공학박사 학위논문

HEVC Rate-Distortion Optimization을 위한 Fast Algorithms

Fast Algorithms
for HEVC Rate-Distortion Optimization

2013년 8월

서울대학교 대학원
전기 컴퓨터 공학부
이 규 중

HEVC Rate-Distortion Optimization을 위한 Fast Algorithms

Fast Algorithms for HEVC Rate-Distortion Optimization

지도교수 이 혁 재

이 논문을 공학박사 학위논문으로 제출함

2013년 8월

서울대학교 대학원

전기 컴퓨터 공학부

이 규 중

이규중의 공학박사 학위논문을 인준함

2013년 8월

위 원 장 : 채 수 익 (인)

부위원장 : 이 혁 재 (인)

위 원 : 조 남 익 (인)

위 원 : 최 기 영 (인)

위 원 : 이ii 채 은 (인)

초록

디지털 영상 기기의 발전과 더불어, 고화질 영상에 대한 수요 또한 함께 증가하고 있다. 최근의 스마트폰과 태블릿 PC의 급속적인 성장은 이러한 추세를 가속화 시키고 있다. 이러한 변화에 맞추어, 고화질 영상 압축을 위한 새로운 영상 압축 기술의 표준화가 ISO/IEC MPEG과 ITU-T/VCEG의 공동의 팀으로 진행되어 왔다. HEVC는 H.264/AVC의 뒤를 잇는 차세대 영상 압축 표준 기술로서, 2013년 1월 FDIS (Final Draft International Standard)가 작성되면서, 표준화 과정이 완료되었다. HEVC는 H.264/AVC 대비 같은 화질의 영상을 절반의 비트량으로 압축하는 것을 목표로 하였으며, 이런 목표를 달성하기 위해, 새로운 기술들이 제안되었다. 특히, 복잡한 block 구조와 크게 늘어난 mode의 수는 영상 압축의 효율을 향상시키는 데에 크게 기여를 하였고, 이는 최적의 mode를 결정하는 RDO (Rate-Distortion Optimization)가 더욱 중요한 역할을 하도록 만들었다. 그러나, 복잡해진 block 구조는 RDO의 연산량 또한 크게 증가시켰다. 이러한 이유로, H.264/AVC와 달리 HEVC에서는 RDO의 연산량을 줄이면서 압축 효율을 유지하는 것이 중요한 이슈가 되었다.

본 논문에서는, H.264/AVC와 HEVC에서의 RDO에 의한 RD 저하의 차이를 실험 결과를 통해 제시하여 문제를 정의하고, RDO의 연산량을 줄이는 알고리즘들을 세 가지 연구 방향을 통해 제안하였다. 첫 번째 방향의 연구에서는 RDO의 과정을 구성하는 Transform, Quantization, Inverse

Quantization, Inverse Transform 그리고 Entropy Coder 등의 일련의 과정의 연산을 단순화하는 알고리즘들이 제안되었다. 이러한 알고리즘은 기본적으로 H.264/AVC에서 이루어진 연구를 기반으로 하였고, 기존 알고리즘의 한계 또한 분석되어 성능을 향상시켰다. 더 나아가서는, 좀 더 공격적으로 RDO의 연산량을 줄일 수 있는 새로운 방법을 제안하였다. 두 번째 방향의 연구에서는 Zero Block detection이라는 기술을 기반으로, HEVC에 적합하게 RDO의 연산을 줄이는 방법을 제안하였다. H.264/AVC에서 제안되었던 알고리즘들은 HEVC에서의 Zero Block을 특징을 제대로 반영하지 못하기 때문에, 단순 수정을 통해 HEVC에 적용할 경우 기대한 만큼의 성능을 얻을 수 없다. 이러한 한계점을 해결하여 HEVC에 적합한 효율적인 Zero Block detection 알고리즘이 제시되었다. 세 번째 방향의 연구에서는, SATD 기반의 RDO를 활용하여, SSE 기반의 RDO의 연산량을 줄이는 방법을 제안하였다. SATD 기반의 RDO와 SSE 기반의 RDO의 차이점 분석과 실험 결과 바탕으로 효율적으로 SATD 기반의 RDO을 활용하는 방법이 제시되었다. 이렇게 제안된 알고리즘들은 HEVC의 reference software인 HM에 구현되어, RDO의 연산량을 크게 줄이면서도, RD 저하가 크게 증가하지 않는 실험 결과를 보이고 있다.

주요어 : HEVC, RDO, Simplified CABAC, Simplified SSE, Zero block detection, Pseudo zero block, SATD upper bound, SATD based RDO

학 번 : 2009-30926

목차

초록 ..... **iii**

목차 ..... **v**

표 목차 ..... **viii**

그림 목차 ..... **x**

제 1 장 서론 **1**

1.1 연구 배경 1

1.2 연구 내용 3

1.3 논문 구성 6

제 2 장 배경지식과 이전 연구 **7**

2.1 배경지식 7

2.2 이전 연구 15

제 3 장 Simplified RDO **19**

3.1 Simplified SSE 19

3.2 Simplified CABAC 24

3.2.1 CABAC 의 구조 24

3.2.2 Various Complexity CABAC 25

3.2.2.1 High-Complexity CABAC 25

3.2.2.2 Medium-Complexity CABAC 26
3.2.2.3 Low-Complexity CABAC 22
3.2.2.4 Evaluation of Various Complexity CABAC 29
3.2.3 Low-Complexity CABAC for HEVC 30
3.3 Advanced Simplified SSE & CABAC 37
3.3.1 Threshold Algorithm 37
3.3.2 Simplified SSE & CABAC without Transform 41
3.4 Evaluation 48

제 4 장 Zero Block Detection 51
4.1 Extension of H.264/AVC Zero Block Detection for HEVC 51
4.1.1 Characteristics of the zero blocks in HEVC 51
4.1.2 ZB detection by an extension of the H.264/AVC algorithm 54
4.2 Zreo Block Detection for HEVC 59
4.2.1 GZB Detection for 16x16 and 32x32 transforms 59
4.2.2 Relaxed conditions for PZB detection 62
4.2.3 Further complexity reduction with SAD(or SATD) test 65
4.2.4 Proposed ZB detection for HEVC 72
4.3 Evaluation 74

제 5 장 SATD based RDO EVALUATION 84
5.1 Difference between SSE based RDO and SATD based RDO 84
5.2 SATD based RDO Evaluation for HEVC 88
5.3 Evaluation 93

제 6 장 결론 95

참고문헌 100

ABSTRACT 105

표 목차

표 1.1 RDO on/off 시 H.264/AVC 와 HEVC 의 BD-BR 증가 비교 2

표 3.1 Simplified SSE 적용시 BD-BR 의 증가 23

표 3.2 Various Complexity CABAC 적용시 BD-BR 의 증가..... 30

표 3.3 Original 과 Modified Low complexity CABAC 적용시 BD-BR 의 증가 비교 33

표 3.4 Only N_s Method 과 Modified Method 적용시 BD-BR 의 증가 비교 .. 36

표 3.5 Threshold Algorithm 적용시 BD-BR 의 증가 비교 39

표 3.6 Threshold Algorithm, Simplified SSE, Various complexity CABAC 적용시 BD-BR 의 증가 비교..... 40

표 3.7 Threshold Algorithm 과 (3-20)의 알고리즘의 BD-BR 의 증가 비교. 44

표 3.8 QP 에 따른 slope(QP, W_{TR}) 46

표 3.9 Simplified SSE and CABAC without Transform 의 DCT 별 적용시 BD-BR 의 증가 47

표 3.10 줄어든 연산량과 RD 저하 측정을 위한 실험의 Simplified RDO 기술 48

표 4.1 전체 ZB 대비 PZB 의 비율 53

표 4.2 (4-12)와 (4-15)에 의해 결정되는 PZB 와 GZB 의 상대적 비율 64

표 4.3 기존 알고리즘과 새로 제안된 알고리즘의 성능 비교 76

표 4.4 기존 알고리즘과 새로 제안된 알고리즘의 적용시, BD-BR 증가와 Time Saving 비교 78

표 4.5 섹션 4.2.3 의 Upper-Bound 적용시 알고리즘 수행시간비 81

표 4.6 ZB detection algorithm 을 RDO 와 FME 에 적용시 BD-BR 의 증가와 전체 encoding 시간 대비 Time Saving 82

표 4.7 RDOQ 를 On 하고, ZB detection algorithm 적용시, BD-BR 의 증가와 Time Saving 83

표 5.1 SSE-RDO 를 SATD-RDO 로 그룹별 대체시 발생하는 BD-BR 의 증가 90

표 5.2 Merge Mode 의 SATD-RDO 대체와 식(5-1), (5-2) 누적 적용시 BD-BR 의 증가와 Time Saving 93

표 6.1 제안된 조합 알고리즘과 ECU 적용시 BD-BR 증가와 Time Saving 98

표 6.2 제안된 조합 알고리즘과 RDO off 적용시 BD-BR 증가와 Time Saving 99

그림 목차

그림 2.1 RDO 과정 7
그림 2.2 HEVC 의 CTU 와 CU 의 Quadtree 구조 9
그림 2.3 HEVC Inter-prediction PU 종류 10
그림 2.4 HEVC CU, PU, TU 간의 관계 11
그림 2.5 HEVC 의 최적 CU depth 결정 과정 12
그림 3.1 주요 신호들이 표기된 RDO 과정 19
그림 3.2 SSE 와 SSE'과의 관계 22
그림 3.3 CABAC 의 과정 24
그림 3.4 Signifcance Map 에 대한 실제 Bits 와 예측 Bits 와의 관계 31
그림 3.5 N_s 와 대응되는 평균 $Level_{abs}$ 과 N_z 34
그림 3.6 N_s 와 32x32 DCT 에 대한 실제 평균 Bits 와의 관계 35
그림 3.7 Threshold Level 변화에 따른 평균 BD-BR 증가와 Skip Ratio..... 41
그림 3.8 식(3-20) 적용시, Real RD cost 와 Estimated RD cost 의 관계..... 43
그림 3.9 N_s 와 SAD_{res} 의 관계..... 45
그림 3.10 다양한 Simplified RDO 기술 조합에 따른 수행시간과 BD-BR 의 증가..... 50
그림 3.11 Condition 4 에 세가지 복잡도 CABAC 을 적용한 실험결과 50
그림 4.1 HEVC 에서의 ZB 결정 과정 52

그림 4.2 32x32 block 의 ZB detection 을 위한 8x8 Hadamard DC coefficients 로 이루어진 4x4 Hadamard transform 60
그림 4.3 32x32 block 에 대한 SAD 대비 $Cost_{null}$ 과 $Cost_{nonnull}$ 65
그림 4.4 HEVC 를 위해 제안된 ZB detection algorithm 73
그림 4.5 기존 알고리즘과 새로 제안된 알고리즘의 수행시간 비 77
그림 4.6 식(4-22)의 C 의 변화에 따른 알고리즘의 성능 변화 79
그림 4.7 식(4-15)의 Threshold 의 변화에 따른 알고리즘의 성능 변화..... 80
그림 5.1 SATD 기반 RD cost 와 SSE 기반 RD cost 의 관계 86

제 1 장 서 론

1.1 연구 배경

HEVC[6]는 H.264/AVC[7]과 비교하여 같은 화질의 영상을 절반의 Bits량으로 압축하는 것을 목표로 표준화 과정이 진행되어 왔고, 2013년 1월 FDIS를 발표함으로써 표준화 과정이 완료되었다. 압축효율의 두 배 향상을 위해, HEVC에는 복잡한 block structure, 크게 늘어난 Intra prediction mode, 복잡해진 interpolation filter 그리고 SAO(Sample Adaptive Offset) 등의 새로운 기술들이 추가되었다. 복잡한 block structure와 크게 늘어난 Intra prediction mode는 최상의 RD(Rate-Distortion) 성능을 제공하는 최적 mode의 선택을 어렵게 하였다. 따라서, HEVC에서 목표로 하는 압축효율 향상을 위해 최적 mode를 선택하는 RDO(Rate-Distortion Optimization)[8]는 매우 중요한 과정임을 알 수가 있다. H.264/AVC에 비해 HEVC에서 RDO가 RD 성능에 미치는 영향을 얼마나 증가했는지를 확인하기 위해 다음과 같은 실험을 진행하였다. 똑같은 영상에 대해서, H.264/AVC에 대해 RDO를 on/off 하여 BD-BR [11]의 증가를 측정하였고, HEVC에 대해 RDO를 on/off 하여 BD-BR의 증가를 측정하였다. H.264/AVC의 경우 RDO를 encoding configuration 파일에서 쉽게 on/off 할 수 있는 데에 반해,

HEVC는 RDO를 off하는 옵션이 없다. 따라서, H.264/AVC의 RDO off 상태와 비슷하게, SSE 기반의 RD cost를 SATD 기반의 RD cost로 모두 대체하여 HEVC에서의 RDO off 상태를 구현하였다. 아래 표 1.1은 RDO on/off 시 발생하는 BD-BR의 증가를 H.264/AVC 와 HEVC에 대해서 제시하고 있다. H.264/AVC의 BD-BR의 증가가 평균 4.25% 인 데 반해, HEVC의 BD-BR의 증가는 평균 15.51%로 4배 가까이 증가함을 알 수 있다. 그러므로, HEVC에서의 RDO는 HEVC가 목표로 하는 압축 효율 향상에 필수적인 과정임을 알 수 있다. 그러나, HEVC에서는 다양한 크기의 transform을 지원하기 때문에, RDO의 연산량 또한 크게 증가하였다. 따라서, 본 논문은, HEVC의 RDO의 연산량을 줄이면서도 RD 저하를 크게 증가시키지 않는 알고리즘들을 제안하고 있다. 기본적으로는 H.264/AVC에서 제안되었던 알고리즘들을 단순 확장하여 HEVC에 적용하여 그 성능을 평가하였고, 그 한계점을 분석하였다. 더 나아가, 이러한 한계점을 기반으로 이를 극복할 수 있는 알고리즘을 제시하였다.

표 1.1 RDO on/off시 H.264/AVC와 HEVC의 BD-BR 증가 비교

	BD-BR	
Class C	H.264/AVC	HEVC
RaceHorses	4.87%	13.45%
BQMall	4.27%	16.33%
PartyScene	3.87%	11.82%
BasketballDrill	3.99%	20.44%
Average	4.25%	15.51%

1.2 연구 내용

본 논문은 HEVC RDO의 연산을 줄이면서 RD 저하를 최소화 할 수 있도록 세가지 방향의 알고리즘을 제안하였다.

첫 번째로는, RDO의 과정들인 DCT(Discrete Cosine Transform), Q(Quantization), IQ(Inverse Quantization), IDCT(Inverse DCT) 와 CABAC (Context-Adaptive Binary Arithmetic Coding) 에 대해서 각각의 과정을 단순화 시키는 기술들을 제안하였다. 기본적으로는 H.264/AVC에서 제안된 기술들을 약간의 수정을 통해 HEVC에 적용하였고, 그에 따른 성능을 분석하였다. 이러한 기술들에 속하는 것은 Simplified SSE(Sum of Squared Errors), Simplified CABAC 이다. Simplified SSE는 잔차 신호와 transform coefficients 간의 수학적인 관계를 분석하여 제안된 기술로 IDCT의 과정을 생략할 수 있는 기술이다. Simplified CABAC은 CABAC의 과정 또는 연산을 단순화 시키는 방법으로, 기존에 제안된 기술들을 세 가지의 복잡도로 구분하여 소개되었고, 그 중 제일 복잡도가 낮은 기술에 대해서는 HEVC에 적합하도록 성능을 향상시켰다. 또한, RDO 연산량을 줄이기 위해 좀 더 공격적인 방법들을 제안하고, 그 방법으로는 Threshold algorithm과 Simplified SSE and CABAC without transform 이 있으며, 두 방법 모두 DCT를 비롯한 이후의 모든 과정을 생략할 수 있는 방법을 제시하고 있다.

두 번째로는, ZB(Zero Block) detection 이라는 DCT를 생략하는

기술로 H.264/AVC에서 제안된 기술의 한계를 분석하였다. DCT는 RDO의 직렬 프로세스의 첫 번째 과정으로써 이후 과정에 필요한 결과를 제공하는 중요 역할을 하고 있다. 이러한 DCT를 생략할 수 있게 되면, 이어지는 과정들의 연산 또한 생략할 수 있기 때문에, RDO의 연산량을 크게 줄이는 것이 가능하다. ZB detection 은 이러한 DCT를 생략하는 효율적인 기술로 잘 알려져 있다. ZB은 quantized transform coefficients 모두가 0인 블록을 지칭하는 용어로, 전송될 encoding data가 필요 없는 블록을 의미한다. 따라서, DCT 과정 이전에 ZB을 정확하게 검출한다면, DCT 포함 그 이후의 프로세스들을 생략할 수 있기 때문에, 해당하는 RDO의 연산량을 줄일 수 있으면서, 동시에 RD 저하를 발생시키지 않을 수 있다. 본 논문에서는 PZB(Pseudo Zero Block)이라는 새로운 개념의 ZB를 정의하고 이를 효율적으로 검출할 수 있는 조건들을 설계하였고, 결론적으로 HEVC에서도 효과적으로 동작하는 ZB detection 알고리즘을 제안하였다. 새로 제안된 알고리즘이 RDO의 연산량을 기존 알고리즘에 비해 훨씬 효과적으로 감소시킬 수 있음을 실험 결과를 통해 제시하고 있다.

세 번째로는, 연산량이 많은 SSE 기반의 RDO를 연산량이 적은 SATD(Sumf of Absolute Transform Difference) 기반의 RDO로 부분적으로 대체함으로써, RDO의 연산량을 줄이는 방법을 제안하였다. SATD 기반의 RDO는 연산량이 적은 만큼 SSE 기반의 RDO의 비해 최적의 mode를 결정하는 때, RD 저하를 초래하게 된다. 그러나, SATD 기반의 RDO로 최적의 mode를 결정해도 RD 저하가 크게 증가하지 않는

조건이 존재하기 때문에, 이 점을 활용하여, 연산량을 줄이면서 RD 저하가 크게 증가하지 않도록 하는 SATD기반의 RDO를 효과적으로 사용하는 방법을 제안하였고, 이를 실험 결과로 제시하고 있다.

이상의 제안된 기술들을 서로 배타적인 기술들이 아니기 때문에 조합이 되어 적용할 수 있다. 본 논문의 결론에서는 RDO의 연산량을 줄이면서, RD 저하를 최소화하는 세 가지 방향의 기술들 중 제일 효과적인 조합에 의한 실험 결과를 제시하고 있다.

1.3 논문 구성

본 논문의 나머지는 다음과 같이 구성된다. 2 장에서는 본 논문의 이해를 위해 RDO 의 과정, HEVC 의 block structure, mode 그리고 best mode 결정 과정을 설명하고, 이전 연구들을 소개한다. 3 장에서는 RDO 의 각 과정의 연산을 단순화하는 Simplified RDO 를 설명하고, 실험 결과로 성능을 제시한다. 4 장에서는 HEVC 에 적합한 ZB detection 알고리즘을 제안하여, RDO 과정을 효과적으로 생략할 수 있음을 제시하고, 이에 대해 기존 알고리즘 대비 성능 향상 실험 결과를 보여준다. 5 장에서는 SSE 기반의 RDO 를 연산량이 적은 SATD 기반의 RDO 로 대체하는 알고리즘을 제시하고, 실험 결과로 성능을 제시한다. 6 장에서는, 각 제안 기술에 대해 요약을 하고, 최종적으로 각 기술의 최적의 조합으로 실험 결과를 제시하며, 성능을 평가하고 결론을 맺는다.

제 2 장 배경지식과 이전 연구

2.1 배경지식

RDO는 Rate-Distortion Optimization의 약어로, 영상 압축에서 best mode를 결정하는 데에 사용되는 과정이다. 영상 압축의 성능은 원본 영상을 얼마나 작게 압축할 수 있는 지와 원본 영상을 얼마나 잘 유지할 수 있는 지를 기준으로 평가된다. 전자는 Rate를 의미하고 후자는 Distortion을 의미한다. RDO는 이 두 가지를 함께 고려하여 최상의 영상 압축의 성능을 보여주는 best mode를 결정한다.

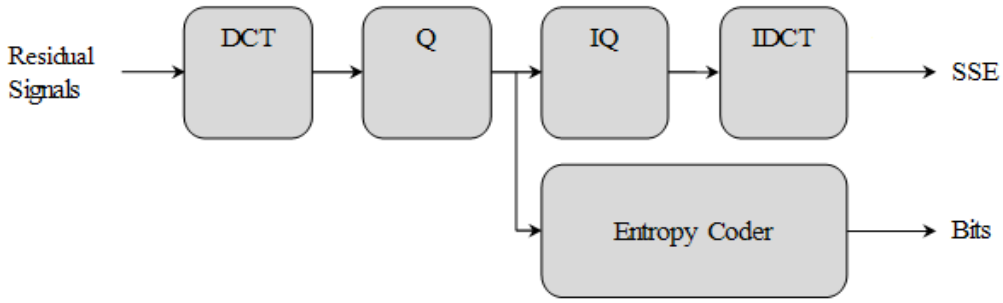


그림 2.1 RDO 과정

RDO 과정은 기본적으로 위의 그림처럼 DCT, Q, IQ, IDCT 그리고 Entropy Coder의 과정을 포함하고 있다. H.264/AVC에서는 Entropy Coder로써 CAVLC (Context-Adaptive Variable Length Coding)와

CABAC(Context-Adaptive Binary Arithmetic Coding)를 모두 사용 가능하였으나, HEVC에서는 최종적으로 CABAC만을 표준기술로 채택하였다. RDO에서는, DCT/Q/IQ/IDCT 를 통해서 SSE(Sum of Squared Errors) 를 계산하고, 이것이 Distortion을 나타낸다. DCT/Q/CABAC 를 통해서 Bits를 계산하고, 이것이 Rate를 의미한다. SSE는 잔차신호와 재구성된 잔차신호의 차이들의 제곱의 합으로 이루어진 값으로, DCT/Q 과정으로 인해서 발생하는 quantization errors의 에너지로 해석될 수 있고, Bits는 DCT/Q의 결과로 발생하는 quantized transform coefficients에 대한 정보를 CABAC으로 압축했을 때의 발생하는 Bits의 양으로 볼 수 있다. 이렇게 계산된 SSE와 Bits 그리고 라그랑지안 계수인 λ_{RDO}을 이용하여 통해 RD cost를 식(1)과 같이 연산한다. λ_{RDO}는 QP(Qunatization Parameter)에 따라 값이 정해지는 상수이다. 결론적으로, RDO는 입력되는 잔차신호에 대해 식(1)에 해당하는 RD cost를 계산하여, minimum RD cost를 발생하는 mode를 best mode로 선택하게 한다.

$$J = SSE + \lambda_{RDO} \times Bits \qquad (2-1)$$

HEVC는 H.264/AVC에 비해 훨씬 복잡한 block structure와 늘어난 Intra prediction mode를 제공한다. 이는 best mode를 선택해야 하는 RDO의 중요성이 커짐을 의미한다. 본 논문의 이해를 위해, HEVC의 block structure과 mode에 대해 설명하고, best mode의 결정 과정을

소개하고자 한다.

HEVC에서 CU(coding unit)는 아래 그림처럼 정사각형 모양으로 QuadTree 구조로 쪼개지게 되어있다. 최상위의 제일 큰 CU를 CTU (coding tree unit)로 표기하고, 그 크기는 64x64, 32x32 그리고 16x16중 하나로 선택이 가능하나, 기본적으로 64x64로 설정된다. QuadTree 구조로 쪼개지기 때문에, depth 라는 개념이 도입되었고, 제일 큰 CU 의 depth는 0이 되고, 제일 큰 CU가 쪼개진 다음 층의 CU의 depth는 1이 된다. 이를 일반화하면 depth가 n인 CU가 쪼개지면 depth가 n+1인 CU가 4개가 생기게 된다.

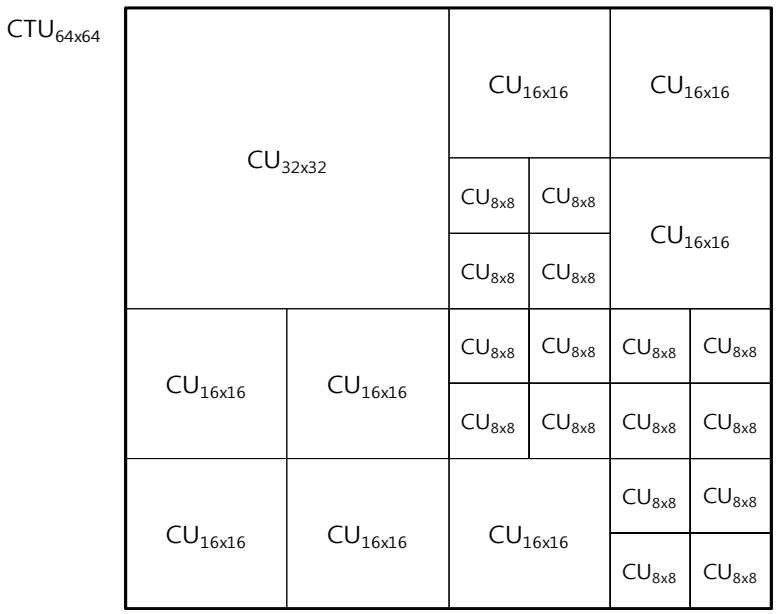


그림 2.2 HEVC CTU와 CU의 Quadtree 구조

한 개의 CU에 대해서 여러 개의 PU(prediction unit)로 나뉘게 된다.

Intra prediction의 경우에는 2Nx2N 과 NxN 두 가지 PU가 존재하며, 각각의 PU에 대해서 prediction mode가 존재하게 된다. 본 논문에서는 이를 Intra mode로 표기한다. Inter prediction의 경우에는, 아래 그림과 같이, 2Nx2N, 2NxN, Nx2N, NxN 과 비대칭 직사각형인 2NxnU, 2NxnD, nLx2N, nRx2N 모두 8개가 존재한다. 그러나, 기본적으로 2Nx2N, 2NxN, Nx2N 만으로 Inter-prediction을 수행한다. 각각의 PU는 자신만의 reference idx와 motion vector를 갖는다.

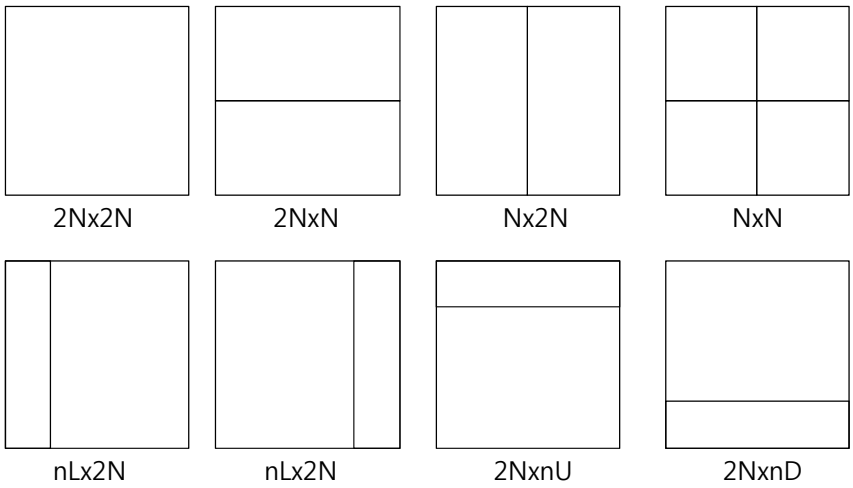


그림 2.3 HEVC Inter-prediction PU 종류

TU(transform unit)은 transform을 수행하는 크기의 단위로 CU와 비슷하게 Quadtree 구조로 쪼개질 수 있다. TU의 크기는 32x32, 16x16, 8x8, 4x4로 다양하며, Inter prediction의 경우 TU가 PU간의 경계에도 걸칠 수가 있다. HEVC에서의 transform은 integer basis

function을 갖는 DCT(discrete cosine transform)이고, Intra prediction의 4x4 transform에 대해서만, 선택적으로 DST(discrete sine transform)가 사용된다.

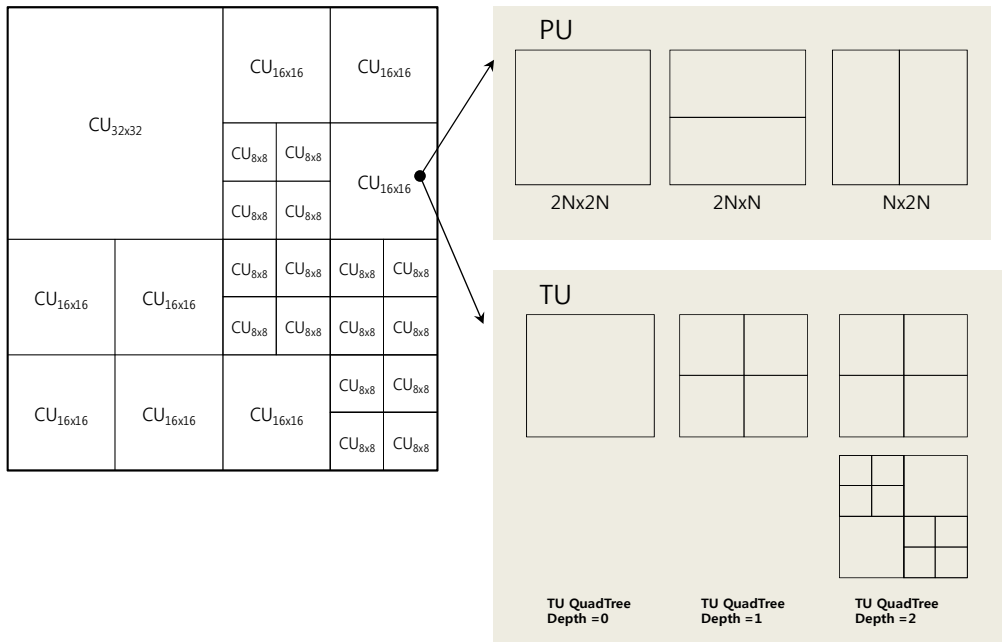


그림 2.4 HEVC CU, PU, TU 간의 관계

위의 그림은 이상의 CU, PU, TU 관계를 보여주는 그림으로, CU, TU는 정사각형 모양만, PU는 정사각형과 대칭 직사각형 모양을 갖는다. PU간의 경계에도 TU가 배치될 수 있기 때문에, PU, TU는 모두 그 최대 크기가 CU 크기에 이해 제한 받게 된다.

HEVC에서 RDO를 통해 best mode를 정한다는 것은, CTU가 어떤 CU로 쪼개지는지, 각 CU가 어떤 PU의 조합으로 이루어지는지 또한 각

CU가 어떤 TU tree 구조를 이루는 지를 정하게 된다. 이를 위해, 아래 그림처럼 CTU를 시작으로 재귀호출 방식을 택하여, 현재 CU의 minimum RD cost와 하위 CU의 minimum RD cost를 비교하여, CU의 쪼개짐을 결정한다. 예를 들어, 첫 번째, 16x16의 현재 CU의 RD cost는 3이고, 하위 4개의 CU의 RD cost의 합은 6이 되어, 16x16 CU는 쪼개지지 않는 것으로 결정된다. 이러한 규칙으로, 64x64 CU의 RD cost가 12 이고 하위 CU에서 재귀 호출되어 결정된 RD cost가 44 이기 때문에, 아래 예로 든 CTU의 경우 depth가 0인 64x64 CU로 최종 encoding 된다.

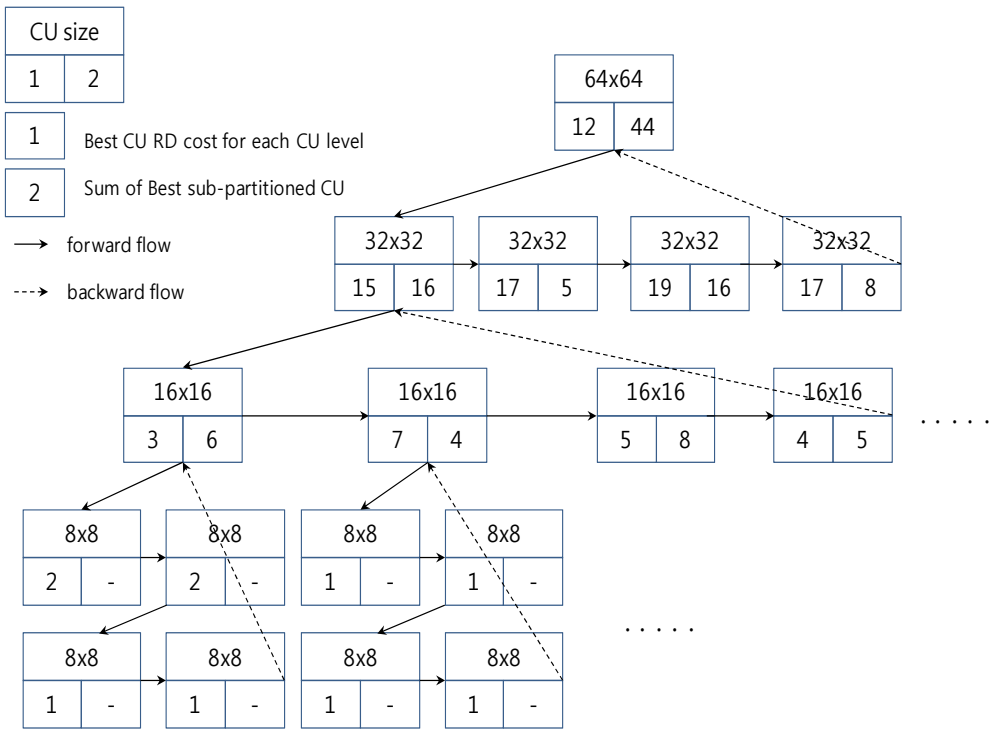


그림 2.5 HEVC 의 최적 CU depth 결정 과정

HEVC에서, CU는 다음과 같은 세 가지, Merge mode, Inter mode Intra mode 하나로 결정된다. Merge mode는 HEVC에서 새로 추가된 mode로써, 공간으로 이웃하는 CU의 motion vector 정보나, 시간 축으로 co-located CU의 motion vector 정보를 그대로 가져다 쓰는 mode로 따로 motion estimation과정을 거치지 않는다. 일반적으로 Merge mode를 위한 motion vector의 후보군의 수는 5개이며 어떤 motion vector를 사용했는지는 merge index로 구분한다. 2Nx2N의 Merge mode에 대해서, encoding될 residual data를 없는 경우에만 Skip mode로 분류되고, 그 외의 경우에는 Inter mode에 포함되는 형식이다. Inter mode는 한 개의 2Nx2N PU, 두 개의 2NxN PU, 두 개의 Nx2N PU 를 제공하며, RD cost의 비교를 통해 CU가 세 중에 하나의 형태로 이루어짐을 결정하게 된다.

Intra mode에서는 H.264/AVC에 비해 훨씬 많은 35개의 mode를 제공하는데 이는 spatial mode가 33개로 크게 증가하였기 때문이다. 이로 인해 Intra mode에 대한 RDO의 연산량이 많이 증가하였기 때문에, Fast Intra 라는 기술을 채택하여 세 단계의 과정을 통해 RDO 연산량을 줄인다. 첫 번째 과정에서는 35개의 모든 Intra mode에 대해서 SATD 기반의 RD cost를 계산하여, 최소의 RD cost를 보이는 N 개의 후보 mode를 선별하고, 주변에 이미 결정된 CU의 best mode를 참조하여, MPM (most probable mode)를 후보 mode에 추가한다. 두 번째 단계에서는, 이렇게 정해진 N+MPM 후보 mode들에 대해서, TU Qaudtree의 maximum depth 을 1로 가정하여 RDO를 수행하여,

minimum RD cost를 제공하는 mode를 최적의 Intra mode로 결정한다. 마지막 세 번째 단계에서는, 앞 단계에서 정해진 Intra mode에 대해서, TU Qaudtree의 maximum depth 을 3으로 두어 최적의 TU 구조까지 결정한다. 이러한 Fast Intra는 encoding시 항상 적용되는 기본 기술로서 본 논문에서는 이 기술이 적용된 상태를 기준으로 연산량의 감소를 측정하였다.

2.2 이전 연구

Simplified RDO의 연구처럼, RDO의 각각의 과정을 단순화하여 전체 RDO의 연산량을 줄이기 위한 많은 연구들이 H.264/AVC에서 진행되었다. 연구[2]에서는 CABAC을 단순화 하기 위해 context-update dependency를 줄이는 방법을 제안하였다. 연구[3]은 연구[2]의 후속 연구로써, 사용되는 context의 수를 줄이고, binary symbol 기반의 병렬화를 통해, CABAC의 연산량을 더 줄이는 방법을 제안하였다. 연구[4]와 연구[14]에서는 bits와 quantized transform coefficients와의 관계를 이용하여, CABAC의 bits를 빠르게 예측할 수 있는 모델링 방법을 제안하였다. 연구[15]에서는 RDO와 CABAC의 복잡도가 분석되고, 하드웨어를 통해, 연산을 크게 줄이는 방법을 제안하고 있다. 연구[1]에서는 SSE를 빠르게 연산하는 방법과 look-up table을 이용한 quantization 방법을 제안하였고, 연구[13]에서는 CAVLC의 bits를 빠르게 예측할 수 있는 모델링 방법을 제안하였다. 이상의 연구들은 H.264/AVC에서 큰 RD 저하 없이 효과적으로 RDO의 연산량을 감소시켰다. 그러므로, 본 논문에서는 이전 연구들 중에 효과적인 알고리즘을 선별하여 HEVC에 적용하여 성능을 평가하였고, 그 한계를 분석하여, 새로운 알고리즘들을 제안하였다.

ZB detection 알고리즘은 transform과 quantization 과정이 있는 영상 압축 기술에 대해 진행되었던 연구여서, 매우 많은 이전 연구들이 존재한다. 본 논문에서는 그 중에서, H.264/AVC 에 대해 제안되었던

기술들을 위주로 소개하도록 한다. 연구[18]에서는 잔차신호의 SAD(sum of absolute difference)의 threshold를 설정하여, 해당 SAD가 설정된 threshold보다 작으면 ZB으로 예측하고, 반대의 경우 ZB이 아닌 것으로 예측한다. 이 threshold는 잔차신호의 SAD와 DCT coefficient의 수학적인 관계를 이용하여 끌어낸 값으로, ZB detection의 정확도가 매우 높은 대신에 ZB의 검출량이 매우 작다. 연구[18]에서 제안된 threshold의 성능을 개선하기 위해, 연구 [26] 와 [27]에서 제안된 ZB detection은 encoding 조건에 따라 threshold를 조절하는 방법을 제안하고 있다. 그러나, 검출량이 상대적으로 작은 편이다. 연구[25]에서는 잔차 신호에 Gaussian distribution으로 모델링과 신호간의 correlation을 이용하여 SAD에 대한 threshold를 정한다. 확률 모델을 사용함으로써, 이전 연구들에 비해 ZB의 검출량이 크게 증가하며, 정확도는 약간 떨어진다. 연구 [25]을 좀 더 개선하기 위해, 연구[22]에서는 Hadamard transform과 DCT의 관계로부터 SATD에 대한 threshold를 제안하고 있다. 연구[17]는 연구[25]과 거의 비슷하나, 잔차신호를 Laplacian distribution을 이용하여 모델링하고 이를 바탕으로 SAD에 대한 threshold을 제안하고 있다. 이상의 확률 모델을 기반으로 SAD의 threshold를 제안하는 연구들은 잔차신호의 주파수적인 특징을 전혀 반영하고 있지 않기 때문에, 특정 잔차신호의 블록에 대해서는 정확도가 매우 떨어질 수 밖에 없는 한계를 보인다. 예를 들어, 같은 SAD을 갖는 두 개의 블록을 생각해보자. 블록 A는 대부분의 에너지가 저주파(혹은 특정 주파수)에 몰려있고, 블록 B는 에너지가 주파수

전체적으로 일정하게 분포한다고 가정해 보자. 이 경우, 블록 B가 블록 A에 비해 ZB가 될 확률이 높아지게 된다. 연구 [19]에서는 잔차신호의 에너지를 DC와 AC로 구분하여 계산하여, 주파수 특징을 고려한 ZB detection을 제안하고 있다. 그러나, 이 방법은 에너지를 구하기 위해, 잔차신호에 대해, 곱셈이 필요한 연산을 추가해야 하기 때문에, 연산량을 줄이기 위한 저복잡도 ZB detection에 적합하지 않다. 연구[20]에서는 Hadamard coefficients를 활용하여 ZB detection을 수행하는 방법을 제안하고 있다. 기본적으로, Hadamard transform과 DCT가 비슷한 basis function을 갖고 있기 때문에, Hadamard coefficients의 활용은 DCT에 의한 주파수 특징을 잘 반영하게 된다. 또한, Hadamard coefficients는 encoding 과정 중 FME(fractional motion estimation)에서 계산되기 때문에, 추가되는 연산 없이 RDO 과정 전에 Hadamard coefficients를 얻을 수 있다. HEVC 에서는 Hadamard transform은 H.264/AVC에서처럼 4x4나 8x8 단위로 수행되고, 16x16과 32x32 에 대한 블록에 대해서는 8x8단위로 쪼개서 수행된다. 이런 이유로 연구[20]에서 제안된 알고리즘을 HEVC에 적용하기 위해서는 16x16과 32x32에 대한 Hadamard transform의 추가적인 연산이 필요하게 된다. 그러나, 이러한 추가되는 연산이 상대적으로 작지 않기 때문에, 본 논문에서 목표로 하는 저복잡도 ZB detection에 적합하지 않다. 또한, 처리하는 블록의 크기가 커짐에 따라, DCT coefficient의 크기에 대한 분산 또한 증가하게 된다. 이러한 현상은 모든 quantized transform coefficients가 0이 아니어도 ZB이 될 수 있는

블록의 출현 비율을 높이게 한다. HEVC에서는 해당 블록이 ZB이라고 가정하여 구한 RD cost가 실제 RDO 과정을 거쳐 구한 RD cost보다 작게 되면, 해당 블록을 ZB으로 처리하여 encoding을 한다. 이러한 종류의 ZB을 본 논문에서는 pseudo ZB으로 정의하고 PZB로 표기하기로 한다. 반면에, 모든 quantized transform coefficients가 0이 되어 ZB의 되는 블록을 PZB와 구분을 위해, genuine ZB으로 정의하고 GZB로 표기하기로 한다. 16x16 이나 32x32의 경우 전체 ZB에 대한 PZB의 비율이 높기 때문에, HEVC를 위한 ZB detection은 이러한 PZB를 잘 검출할 수 있어야 한다. 본 논문에서는 우선 기존 H.264/AVC에서 제안된 알고리즘을 그대로 확장하여 HEVC에 적용하여 새로 제안하는 알고리즘과 비교하였다.

본 논문의 SSE 기반의 RDO를 부분적으로 SATD 기반의 RDO로 대체하는 방법 관련하여 H.264/AVC에 대해 진행된 이전 연구들은 없다. 그 이유로는 H.264/AVC의 경우 RDO on/off 시의 RD 저하가 크지 않았고, 이미 RDO의 각 과정을 단순화하는 기술들과 ZB detection을 통해, RDO의 연산량을 크게 줄이면서, RD 저하를 최소화하는 기술들만으로도 충분히 효과를 낼 수 있었기 때문에, 부분적으로 SATD 기반의 RDO를 사용할 필요가 없었기 때문이라 판단된다. 본 논문에서는 HEVC에서 SSE 기반의 RDO를 부분적으로 SATD 기반의 RDO로 대체하는 방법이 매우 효과적임을 실험 결과를 통해 제시하고자 한다.

제 3 장 Simplified RDO

3.1 Simplified SSE

논문 [1] 에서는 RD cost를 구성하는 Distortion 성분인 SSE를 연산함에 있어서, 수학적인 관계를 이용하여 IDCT를 생략하는 알고리즘을 제안하고 있다. 아래 그림은 RDO 과정을 나타내는 것으로 알고리즘을 설명하기 위한 신호들이 표시되어 있다.

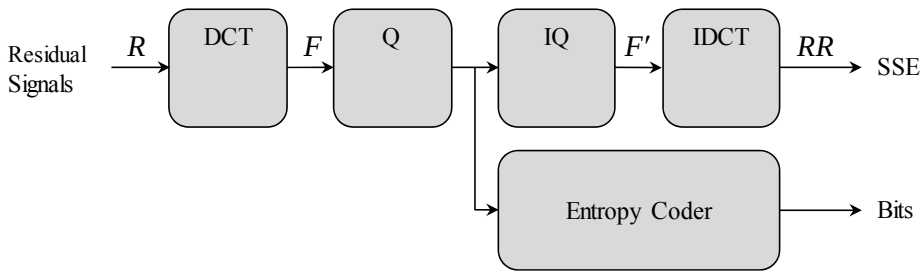


그림 3.1 주요 신호들이 표기된 RDO 과정

알고리즘을 설명하기 신호들은 다음과 같이 정의한다.

S : original signals

RS : reconstructed signals

P : predicted signals

R : residual signals

RR : reconstructed residual signals

F : forward transform coefficients

F' : backward transform coefficients

연구[1]의 알고리즘의 소개를 위하여, RDO의 RD cost식을 쓰면 식(3-1)과 같다. 연구[1]은 Distortion에 해당하는 SSE 연산을 빠르게 하는 방법을 제안하고 있다.

$$J = SSE + \lambda_{RDO} \cdot Bits \qquad (3-1)$$

원신호인 S는 잔차신호 R과 예측신호 P의 합으로, 재구성 신호인 RS는 재구성 잔차 신호 RR과 예측 신호 P의 합으로 아래와 같이 표현된다.

$$S = R + P \qquad (3-2)$$

$$RS = RR + P \qquad (3-3)$$

SSE(Sum of Squared Errors)는 그 정의에 따라 원신호와 재구성 신호의 차이의 제곱의 합으로 식(3-4)와 같이 표현이 된다.

$$SSE = \| S - RS \|^2 \qquad (3-4)$$

이는 식(3-4)에 식(3-2) 와 식(3-3)을 대입하면 식(3-5)와 같이 잔차신호 R과 재구성 잔차신호 RR로 SSE를 표현할 수 있다.

$$SSE = \| R - RR \|^2 \qquad (3-5)$$

이때, DCT에 사용되는 행렬을 T라고 할 때, T가 unitary matrix라면 식(3-5)는 식(3-6)처럼도 표현이 가능하다.

$$SSE = \| T(R - RR)T^T \|^2 \qquad (3-6)$$

이때, HEVC에서의 T는 수학적으로 완벽한 unitary matrix가 아니기 때문에 F와 F′로 구한 SSE′ (식(7))와 S와 RS로 구한 SSE(식(4))는 완벽하게 일치하지는 않는다.

$$SSE' = \| T(R - RR)T^T \|^2 = \| F - F' \|^2 \qquad (3-7)$$

그러나, HEVC에서의 T는 unitary matrix와 매우 유사하기 때문에, 그 값의 차이는 매우 작다. 아래 그림은 x축에 SSE를 y축에 SSE′을 나타낸 것으로, 모든 실험 자료들이 y=x인 추세선에 매우 가깝게 밀집됨을 확인할 수가 있다.

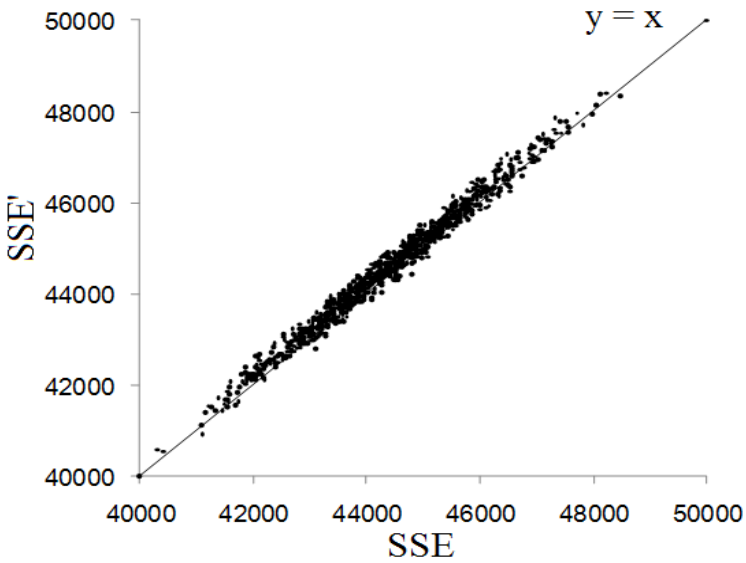


그림 3.2 SSE와 SSE'과의 관계

이렇게 SSE'가 SSE를 정확하게 근사하기 때문에, RD 성능 저하에 미치는 영향이 매우 작다. 이상의 알고리즘은 H.264/AVC를 위해 제안된 것이지만, HEVC에도 쉽게 적용할 수 있으며, IDCT에 해당하는 연산량을 줄일 수 있다. Simplified SSE를 적용했을 때의 RD 저하를 측정하기 위해, Class B (1920x1080) 영상 4개와 Class C (832x480) 영상 4개를 사용하였고, low-delay configuration으로 encoding 하였다. 아래 표 3.1은 Simplified SSE를 적용했을 때의 BD-BR의 증가를 나타내는 표로, 평균 BD-BR의 증가가 0.003%로 매우 작음을 확인할 수 있다.

표 3.1 Simplified SSE 적용시 BD-BR의 증가

Resolution	Sequence	BD-BR increase
480p	RaceHorses	0.04%
	BQMall	0.13%
	PartyScene	0.06%
	BasketBallDrill	-0.14%
1080p	ParkScene	0.06%
	Cactus	0.02%
	BQTerrace	-0.21%
	BasketBallDrive	0.06%
	Average	0.003%

3.2 Simplified CABAC

CABAC은 현재 처리하는 symbol의 결과가 다음 symbol의 CABAC operation에 영향을 주는 symbol 단위의 serial process이기 때문에, 연산량을 줄이기가 쉽지 않다. 그러나, RDO 과정에서의 CABAC은 생성되는 bitstream의 크기로만 RD cost를 계산하기 때문에, bitstream의 내용은 달라져도 bitstream의 크기만 정확하게 예측할 수 있다면, RD cost도 정확하게 예측할 수 있다. H.264/AVC에서는 [2]~[4]의 연구들이 이러한 원리를 기반으로 CABAC의 연산량을 줄이는 방법을 제안하였다. 본 논문에서는 CABAC의 구조에 대해서 간략히 설명한 후, [2]~[4] 의 연구를 복잡도에 따라 세 가지로 분류하고, 적절한 수정을 통해 HEVC에 적용하였으며, 그에 따른 RD 저하를 제시하고 있다.

3.2.1 CABAC 의 구조

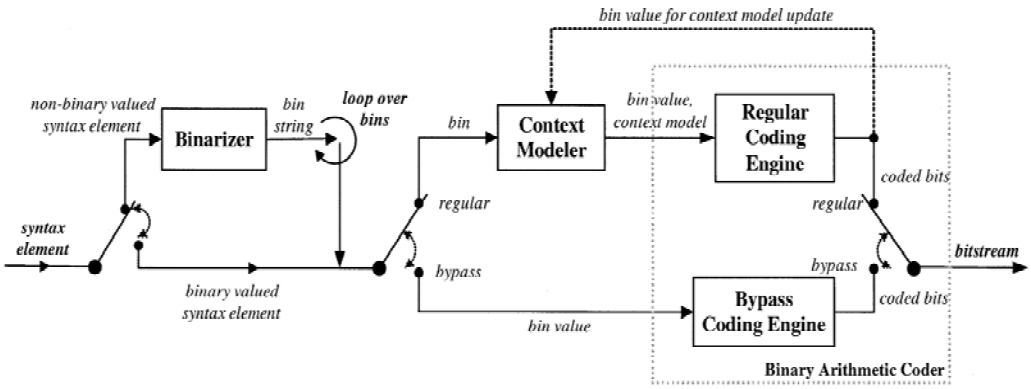


그림 3.3 CABAC의 과정

CABAC는 Context-Adaptive Binary Arithmetic Coding의 약어로, 주변 정보를 활용하는 BAC(Binary Arithmetic Coder)로 볼 수 있다. 위의 그림은 CABAC의 구조와 처리 과정을 나타내고 있다. CABAC은 크게 Binarizer, Context Modeler, 와 BAC로 나뉘어 지며, 입력으로는 syntax element를 받고, 출력으로는 bitstream을 생성한다. CABAC 의 입력인 syntax element는 motion vector, quantized coefficients, prediction mode 등이 될 수 있다. Binarizer는 입력된 syntax element 중 non-binary syntax element에 대하여 이진화를 수행한다. 이진화가 된 bin 은 syntax element의 종류에 따라 Regular Coding Engine을 사용할 지 Bypass Coding Engine을 사용할 지 결정하게 된다. Regular Coding Engine을 사용하는 경우, Context Modeler에서 선택한 context를 사용하여 encoding 되며, Bypass Coding Engine을 사용하는 경우, context를 사용하지 않고 encoding하게 된다. CABAC 에 syntax element가 입력되면 가장 먼저 Binarizer에 의해 이진화 과정을 수행한다. 이 때 생성된 bin의 특성에 따라 Context Modeler가 사용할 context를 선택하게 된다. 이 때 선택된 context로 Binary Arithmetic Coder에서 encoding을 수행하고 context를 Update하는 과정을 거친다.

3.2.2 Various Complexity CABAC

3.2.2.1 High-Complexity CABAC

[2]에서의 연구는 기존 CABAC에 두 가지의 단순화 방법을 제안하고

있다. 첫 번째 방법으로는 CABAC의 BAC(Binary Arithmetic Coding)를 look-up table로 대체하여 renormalization loop 과정을 생략한다. 두 번째 방법으로는 context modeler에서 사용 빈도가 높은 context만을 선별하는 방식으로 실제 사용되는 context의 개수를 줄여서 연산량을 감소시킨다. 이를 HEVC에 적용하기 위해, 실제 사용 빈도가 높은 context를 실험적으로 구하고, 새로 정해진 context에 맞춰서 context modeling 과정을 수정하였다. 이 방법으로, significant map에 사용된 context의 개수를 48에서 26으로, level 에 사용된 context의 개수가 56에서 12로 각각 줄였다.

3.2.2.2 Medium-Complexity CABAC

연구[3]은 연구[2]를 기반으로, 여러 개의 bin에 대한 context update를 한 번에 처리할 수 있는 방법을 제안하고 있다. 기존의 CABAC이 매 bin에 대해서 context update를 하는 것에 반해, 제안된 방법은 여래 개의 bin을 그룹화하여 한 번만 context update를 하기 때문에 연산량이 줄어들게 된다. 이상의 알고리즘을 적용하여 최종 예측하고자 하는 Bits를 예측하기 위해, 기본적으로 다음과 같은 3 단계를 거치게 된다. 첫 번째로는 같은 contexts를 사용하는 bin들의 그룹을 만든다. 따라서, bin들에 의해 사용되는 context의 개수와 bin들로 이루어진 그룹의 개수가 같아지게 된다. 두 번째로는, MPS(most probable symbol)이 개수와 LPS(least probable symbol)의 개수를 첫 번째 단계에서 만든

그룹별로 센다. 세 번째로는, 두 번째 단계에서 구한 MPS와 LPS의 개수를 기반으로 발생되는 Bits를 예측한다. 이 알고리즘의 성능은 Bits를 예측하는 데에 사용되는 그룹의 bin의 개수를 조절함으로써 조정이 가능하다. 그룹의 bin의 개수를 L_{max} 로 표기하면, L_{max} 가 증가함에 따라, 연산량을 많이 줄일 수 있지만 RD 저하 또한 증가하게 된다. 반대로, L_{max} 가 감소하면, 연산량은 적게 줄어들지만, RD 저하는 감소하게 된다. 이상의 알고리즘을 HEVC에 적용하기 위해서는, bin을 위에 설명된 것처럼 그룹화하여야 한다. 이를 위해, last significant XY를 위해서는 58개의 그룹을, significant map을 위해서는 26개의 그룹을, level을 위해서는 12개의 그룹을 발생시켰다. 이 각 그룹에 대한 MPS와 LPS의 개수가 세어지고, 이를 기반으로 Bits를 예상하였다. L_{max} 는 RD저하를 최소화면서 연산량을 줄일 수 있도록 실험적으로 정하였으며, 본 논문에서는 L_{max} 를 16으로 정하여 성능을 측정하였다.

3.2.2.3 Low-Complexity CABAC

앞서 소개된 연구들은 CABAC의 기본적인 과정을 두고 각 과정을 빠르게 처리할 수 있는 방법을 제안하였다면, 연구 [4]에서는 CABAC의 과정을 배제하고, zero 와 non-zero quantized transform coefficients 그리고 quantized transform coefficients의 level를 활용하여 Bits를 예측한다. 이 알고리즘을 설명하기 위해, 다음과 같은 신호들을 정의한다.

R_{Sign}: bits for sign of coefficients

$R_{Significance\ Map}$: bits for significance map

R_{Level}: bits for the absolute value of a coefficient

$R_{Coeff\_abs\_level}$: sum of R_{Level} for all coefficients ($\sum R_{Level}$)

N_s: the number of non-zero (significant) coefficients

N_z: the number of zero coefficients before the last non-zero coefficient

R_{Sign} 의 경우, 실제 CABAC에서도 bypassed path에 의해 encoding 되기 때문에, R_{Sign} 에 대해 발생하는 Bits는 식(3-8)과 같이 표현된다. 또한, significance map에 대해 발생하는 Bits는 N_s 와 N_z 의 합에 비례하는 특징을 가졌기 때문에, 식(3-9)와 같이 표현된다. 식(3-9)에 있는 weighting fact인 1과 1.5는 실험적으로 정해졌다.

$$R_{Sign} = N_s \tag{3-8}$$

$$R_{Significance\ Map} = 1 \times N_z + 1.5 \times N_s \tag{3-9}$$

coefficients의 level 이 5보다 작은 경우, 발생하는 Bits의 량과 coefficients의 level이 직접적으로 비례하기 때문에, 이 경우 coefficients의 level로 발생하는 Bits량을 대체한다. 그 이외의 경우, coefficients의 level로 Bits량을 예측할 수 없기 때문에, 이전 frame에서 CABAC의 결과로 현재 처리하려는 Bits량을 대체한다.

이상의 R_{Level} 에 대한 Bits 예측은 식 (3-10)에 정리되어 있다.

$$R_{Level} = (Level_{abs} \leq 5)\ ?\ Level_{abs} :\ Previous\ CABAC\ Result \qquad (3-10)$$

($Level_{abs}$: the absolute value of a coefficient)

R_{Sign}, $R_{Significance\ Map}$ 그리고 R_{Level} 에 대한 Bits에 대한 예측식들 (3-8),(3-9),(3-10)을 식 (3-11)과 같이 하나로 합치게 되면, DCT/Q의 결과에 대한 CABAC의 Bits를 대체할 수 있게 된다.

$$R_{estimation} = R_{Sign} + R_{Significance\ Map} + R_{Coeff\_abs\_level} \qquad (3-11)$$

3.2.2.4 Evaluation of Various Complexity CABAC

앞서 설명되었던 세가지 복잡도의 CABAC를 HEVC에 적용했을 때의 RD 저하를 측정을 위한 실험을 수행하였고 그 결과를 아래 표 3.2에 제시하고 있다. RD 저하를 측정하기 위해, Class B (1920x1080)[12] 영상 4개와 Class C (832x480)[12] 영상 4개를 사용하였고, low-delay configuration으로 encoding 하였다. High-Complexity 와 Medium-Complexity CABAC은 BD-BR의 증가가 각각 0.28%와 0.40%로 매우 미미한데 반해, Low-Complexity CABAC은 다소 높은 수준의 3.71%인 BD-BR의 증가를 보이고 있다. Low-Complexity CABAC는 앞서 설명되었던 것처럼, CABAC의 과정을 수행하지 않고

Bits를 예측하기 때문에, 그 복잡도 매우 작음을 고려하면, 해당 RD 저하가 크게 문제가 되지 않을 수 있다.

표 3.2 Various Complexity CABAC 적용시 BD-BR의 증가

		BD-BR increase		
Resolution	Sequence	High	Mid	Low
480p	RaceHorses	0.36%	0.50%	2.75%
	BQMall	0.39%	0.56%	4.37%
	PartyScene	0.62%	0.70%	3.78%
	BasketBallDrill	0.37%	0.45%	3.37%
1080p	ParkScene	0.42%	0.55%	3.42%
	Cactus	0.29%	0.53%	4.09%
	BQTerrace	-0.24%	-0.17%	0.94%
	BasketBallDrive	0.02%	0.08%	6.99%
	Average	0.28%	0.40%	3.72%

3.2.3 Low-complexity CABAC for HEVC

앞선 소개된 세 가지 복잡도의 CABAC 중 Low-complexity CABAC은 CABAC의 기본 과정을 거치지 않기 때문에, 연산을 크게 줄일 수 있고, 소프트웨어뿐만 아니라 하드웨어 구현에 매우 용이하다. 그러나, 다른 복잡도의 CABAC에 비해 RD 저하가 크기 때문에, HEVC에 맞게 성능을 향상시키는 방법을 제안하고자 한다.

우선 R_{Sign} 의 경우 실제 CABAC에서도 bypassed path로 구해지기 때문에 정확하게 N_s 와 일치하여 정확한 예측이 가능하다. 두 번째로, $R_{Significance\ Map}$ 의 경우 정확도를 높이기 위해 HEVC의 특징을 반영하도록

한다. HEVC는 다양한 크기의 DCT를 제공하기 때문에, DCT 크기 별로 최상의 결과를 주는 weighting factor가 다를 수 있다. 아래 그림 3.4(a)은 32x32 DCT에 대하여, 실제 $R_{Significance\ Map}$ 를 x축으로 하고, 기존의 weighting factor를 적용한 $R_{Significance\ Map}$ 를 y축으로 한 실험 결과를 보여주고 있고, 그림 3.4(b)는 새로운 weighting factor를 적용한 $R_{Significance\ Map}$ 를 y축으로 한 실험 결과를 보여주고 있다. 새로운 weighting factor를 적용했을 때, 실제 $R_{Significance\ Map}$ 에 보다 가까워짐을 시각적으로 확인할 수 있다.

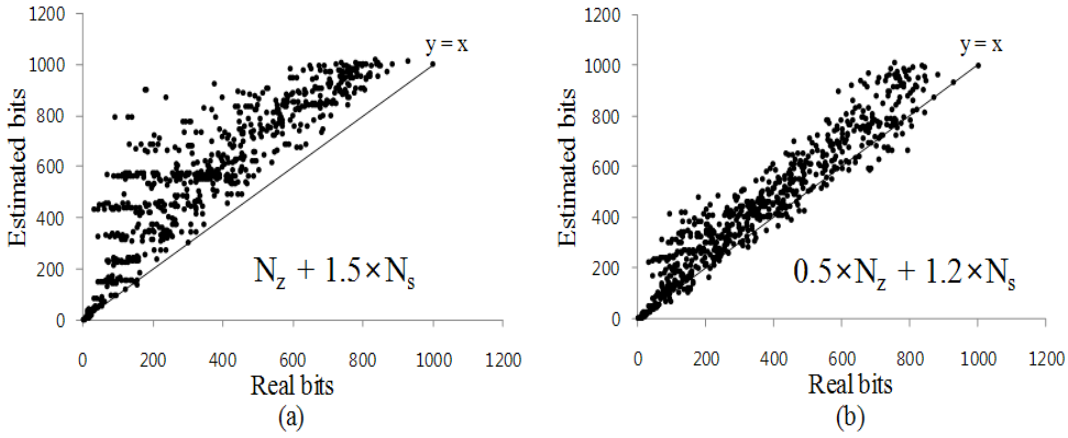


그림 3.4 Significance Map에 대한 실제 Bits와 예측 Bits와의 관계: (a) 32x32 DCT에 이전 방법 적용시 (b) 32x32 DCT에 수정된 방법 적용시

이러한 방식으로 DCT 크기 별로 y=x 에 대한 error을 최소화할 수 있는 weighting factor를 실험적으로 구하였으며, 이 실험에는 Class C 영상의 초반 20 frame의 encoding 결과가 통계 자료로 이용되었다.

이렇게 정해진 DCT 크기별 weighting factor는 식(3-12)에 정리되었다.

$$
\begin{aligned}
R_{Significance\ Map} &= 0.5\times N_z + 1.2 \times N_s \quad for\ 32\times32 \\
&= 0.5\times N_z + 1.5 \times N_s \quad for\ 16\times16 \\
&= N_z + 1.3 \times N_s \quad for\ 8\times8 \\
&= N_z + 1.2 \times N_s \quad for\ 4\times4
\end{aligned}
\tag{3-12}
$$

끝으로, R_{Level} 에 정확도를 향상 시키기 위해, 아래 그림과 같이 coefficient level에 따른 평균 발생 bits를 분석하였다. 또한, 기존 연구의 경우 coefficient level이 5 이상일 때, 이전 프레임의 CABAC의 결과를 사용하기 때문에, 계속적으로 평균을 계산하고 그 결과를 저장하기 때문에 실제 구현시 복잡도가 커지게 된다. 이러한 이유로, 본 논문에서는 아래 그림의 그래프를 구간으로 나누어 선형 근사하는 방식으로 접근하였다. 이러한 구간별 선형 근사의 결과는 식(3-13)에 정리되어 있다.

$$
\begin{aligned}
&\text{if } (\text{Level}_{abs} <= 1) \quad R_{Level} = 0; \\
&\text{else if}(\text{Level}_{abs} <= 2)\ R_{Level} = 2; \\
&\text{else if}(\text{Level}_{abs} <= 3)\ R_{Level} = 5; \\
&\text{else if}(\text{Level}_{abs} <= 4)\ R_{Level} = 5; \\
&\text{else if}(\text{Level}_{abs} <= 5)\ R_{Level} = 6; \\
&\text{else if}(\text{Level}_{abs} <= 15)\ R_{Level} = 0.4\times \text{Level}_{abs} + 6;
\end{aligned}
\tag{3-13}
$$

else if($Level_{abs}$ <= 90) R_{Level} = 0.14× $Level_{abs}$ + 9;

else if($Level_{abs}$ <= 200) R_{Level} = 0.03× $Level_{abs}$ + 20;

else R_{Level} = 0.008× $Level_{abs}$ + 25;

이상의 알고리즘을 적용하여, RD 저하를 측정하여 아래 표에 보여주고 있다. 기존의 방법에 의한 BD-BR의 증가가 평균 3.71%에 반해, 새로 제안된 방법에 의한 BD-BR의 증가는 평균 2.40% 로 개선되었다.

표 3.3 Original 과 Modified Low complexity CABAC 적용시 BD-BR의 증가 비교

Resolution	Original Method	Modified Method
480p	3.86%	2.35%
1080p	3.57%	2.45%
Average	3.71%	2.40%

새로 제안된 low-complexity CABAC은 N_z, N_s 그리고 $Level_{abs}$ 의 세 가지의 값을 활용하여 Bits를 예측한다. 그러나, 이 세가지 값에 적용되는 weighting factor들이 모두 실험적으로 정해지기 때문에, 최적의 값을 찾기가 쉽지 않다. 이러한 이유로 본 논문에서는, N_s 로만 Bits를 예측하는 low-complexity CABAC를 제안하고자 한다.

이를 위해 다음과 같은 quantized transform coefficients의 특징을 다시 고려해보도록 한다. 대부분의 quantized transform coefficients은 0에 가깝고, non-zero quantized transform coefficients 의 경우도 그

크기가 크지 않다. 그래서, N_s 와 Level_{abs} 는 양의 상관관계를 갖게 된다. 이는 아래 그림 3.5(a)를 통해 실험적으로 확인할 수 있다. 이 그림에서 x축은 N_s 를 나타내며, y축은 N_s 에 대응되는 Level_{abs} 의 평균값을 나타낸다. 그림 3.5(b)에서는 N_s 와 N_s에 대응되는 평균 N_z 의 관계를 보여주고 있는데, 여기도 비슷한 경향을 발견할 수 있다. 이러한 그림 3.5 의 실험적인 경향으로부터, 식(3-11)을 새로운 파라미터인 $K(W_{TR})$을 사용하여 식 (3-14)와 같이 수정할 수 있다. W_{TR} 은 Transform의 크기를 나타낸다.

$$\mathrm{R}_{estimation} = K(W_{TR}) \times \mathrm{N_s} \qquad (3-14)$$

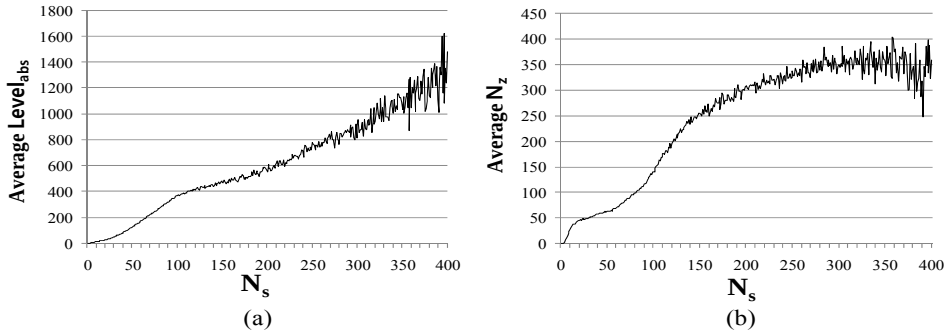


그림 3.5 (a) N_s 와 대응되는 평균 Level_{abs} (b) N_s 와 대응되는 평균 N_z

$K(W_{TR})$ 를 결정하기 위해, 480p 영상 초반 20 frame의 결과가 통계 데이터로 이용되었으며, 그림 3.6은 이 따른 실험 결과의 일부를 시각적으로 보여주고 있다. 그림 3.6의 x축은 N_s 를 나타내고, y축은 32x32 DCT에 대해서 실제 발생한 평균 bits의 수를 나타낸다. 이러한

실험 결과를 바탕으로, $K(W_{TR})$은 32x32, 16x16, 8x8, 4x4 DCT 에 각각에 대해, 4.31, 4.63, 5.5, 5.13으로 결정되었다.

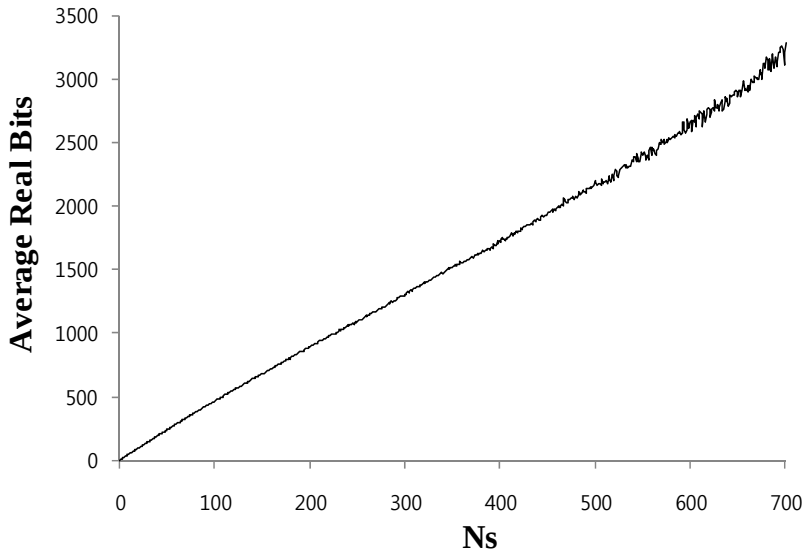


그림 3.6 N_s 와 32x32 DCT에 대한 실제 평균 Bits와의 관계

N_s만의 low-complexity CABAC의 성능을 평가하기 위해, 식(3-12)와 식(3-13)을 적용한 low-complexity CABAC과 RD 저하를 비교하였다. 아래 표 3.4에서 N_s만의 low-complexity CABAC을 'Only N_s Method' 로 표기하였고, 식(3-12)와 식(3-13)을 적용한 low-complexity CABAC을 'Modified Method'로 표기하였다. Only N_s Method 는 평균 1.92% BD-BR의 증가를 보이는 데 반해, Modified

Method는 평균 2.40% BD-BR의 증가를 보인다. Only N_s Method의 BD-BR의 증가가 다소 작지만, 그 차이는 크지 않다. 그러나, Only N_s Method 는 weighting factor를 실험적으로 구하기가 보다 용이하고, 이후 제안되는 알고리즘에서 RDO 연산을 크게 줄이는 중간 단계의 과정으로 사용된다. 이러한 이유로, 본 논문에서는 Only N_s Method를 HEVC를 위한 Low-Complexity CABAC로 사용하기로 한다.

표 3.4 Only N_s Method과 Modified Method 적용시 BD-BR의 증가 비교

Resolution	Modified Method	Only Ns Method
480p	2.35%	1.84%
1080p	2.45%	2.00%
Average	2.40%	1.92%

3.3 Advanced Simplified SSE & CABAC

이상의 H.264/AVC에서 제안된 알고리즘과 성능 향상을 위해 수정된 알고리즘들 모두 DCT/Q의 과정을 꼭 거쳐야 하기 때문에, 연산량을 줄이는 데 근본적인 한계가 있다. 본 논문에서는 좀 더 공격적으로 연산량을 줄일 수 있는 방법을 두 가지 형태로 제안하고자 한다.

3.3.1 Threshold Algorithm

알고리즘을 제안하기 위해, m x m (= n) 크기의 transform에 대해, Simplified SSE를 식(3-15)처럼 다시 적기로 한다. 우선, 대부분의 quantized transform coefficients가 0인 경우를 생각해보자. 이것은 결국 F'1 ~ F'n 의 대부분이 0인 경우를 의미하는 것이기 때문에, SSE'은 식(3-16)의 SSE"로 근사될 수 있다. SSE"는 주파수 영역에서의 에너지를 의미하고, 에너지 보전법칙에 따라 주파수 영역에서의 에너지는 공간 영역에서의 에너지와 같아야 한다. 따라서, SSE"는 식(3-17)의 SSR (Sum of Squared Residuals)과 같은 값을 같게 된다. 따라서, 대부분의 quantized transform coefficients가 0인 경우에 대해서, SSE는 SSR로 대체 될 수가 있다.

$$SSE' = (F1 - F'1)^2 + \cdots\cdots + (Fn - F'n)^2 \qquad (3-15)$$

$$SSE'' = F1^2 + \cdots\cdots + Fn^2 \quad (3-16)$$

$$SSR = R1^2 + \cdots\cdots + Rn^2 \quad (3-17)$$

대부분의 quantized transform coefficients가 0인지 아닌지를 확인하는 검사를 단순화하기 위해, 본 논문에서는 평균 SSR (= SSR/n)을 사용하기로 하고, SSR_{avg} 로 표기하기로 한다. SSR_{avg}를 구해서 미리 정해진 threshold보다 작으면, 대부분의 quantized transform coefficients가 0으로 판단하기로 하고, 이 때에만 SSE를 SSR로 대체한다. 또한, 대부분의 quantized transform coefficients가 0인 경우 발생되는 Bits가 거의 없기 때문에, 0으로 정하기로 한다. 이러한 알고리즘은 아래 식(3-18)에 정리가 되어 있다. SSR_{avg}이 미리 정해진 threshold보다 큰 경우에는, 정상적인 RDO 과정을 거쳐서 계산된 SSE와 Bits를 사용하기로 한다. 이 때, QP에 따라 quantized transform coefficients의 개수가 달라지기 때문에, threshold 또한 QP에 종속되는 변수로, Threshold(QP)로 표기하기로 한다. Threshold(QP)의 값은 모든 테스트 영상에서 BD-BR 증가가 1.0% 미만이 되도록 실험을 통하여 결정되었다. 본 논문에서는 이상의 제안된 알고리즘을 Threshold Algorithm으로 정의한다.

```
if (SSR_avg < Threshold(QP))                                    (3-18)
    SSE = SSR; Bits = 0;
else
    SSE = ‖S - RS‖^2;  Bits = output of CABAC;
```

Threshold(QP)는 테스트 영상에서 BD-BR의 증가가 1.0% 미만이 되도록 실험적으로 정하였다. 이러한 Threshold(QP)를 적용하여, RDO를 수행할 경우, 아래 표 3.5에서 보듯이 BD-BR 의 증가는 평균 0.31%로 매우 작은 수준임을 알 수 있다.

표 3.5 Threshold Algorithm 적용시 BD-BR의 증가 비교

Resolution	Sequence	BD-BR increase
480p	RaceHorses	0.52%
	BQMall	0.34%
	PartyScene	0.04%
	BasketBallDrill	0.33%
1080p	ParkScene	0.32%
	Cactus	0.01%
	BQTerrace	0.27%
	BasketBallDrive	0.73%
	Average	0.32%

제안된 알고리즘의 else 조건문에 해당하는 부분에서는 정상적인 RDO과정을 수행하고 있는데, 이 분에 앞서 제안된 알고리즘들이 쉽게 결합되어 적용될 수 있다. 즉, else 조건에서 SSE는 앞서 제안된 Simplified SSE (SSE')로 대체될 수 있고, Bits는 3.2 섹션에서 제안된 세가지 복잡도의 CABAC에 의해 대체될 수 있다. 이상의 조합으로 RDO를 수행할 경우, 아래 표 3.6와 같은 RD 저하를 얻을 수 있다. Simplified SSE와 High 또는 Medium complexity CABC를 Threshold Method와 결합시킬 경우 Threshold Method만 적용했을 때에 비해,

BD-BR의 증가가 미미하고, Low-complexity CABAC을 결합시키는 경우, Low-complexity CABAC에 의한 BD-BR의 증가분이 더해져서, 평균 2.29%의 BD-BR의 증가를 보이고 있다.

표 3.6 Threshold Algorithm, Simplified SSE, Various Complexity CABAC 적용시 BD-BR의 증가비교

	BD-BR increase		
Options	480p	1080p	Average
Real SSE +CABAC	0.27%	0.35%	0.31%
Simplified SSE + High CABAC	0.37%	0.14%	0.26%
Simplified SSE + Mid CABAC	0.47%	0.26%	0.37%
Simplified SSE + Low CABAC	2.31%	2.26%	2.29%

현재 정해진 Threshold(QP)의 값이 증가하게 되면, 생략되는 DCT의 비율도 증가하지만 RD저하의 정도도 증가하게 된다. Threshold(QP)값의 변화에 따른 줄어드는 연산량과 RD 저하의 trade-off 관계를 분석하기 위해, 정해진 Threshold(QP)를 기준으로 두 배(x2), 세 배(x3)을 하였다. 각 경우에 대해서 생략되는 DCT의 비율과 RD 저하를 측정하였고 아래 그림 3.7에 표시하였다. 생략되는 DCT의 비율은 QP와 영상에 따라 값이 달라지기 때문에 평균값을 사용하였다. Class B (1920x1080) 영상 4개와 Class C (832x480) 영상 4개를 사용되었다. Threshold(QP)의 값이 증가하는 것에 비례하여 생략되는 DCT의 비율과 RD저하의 정도도 빠르게 증가함을 확인할 수 있다. 따라서, RD

저하를 최소화하기 위해서는 Threshold(QP)의 값을 낮은 수준으로 유지해야 한다.

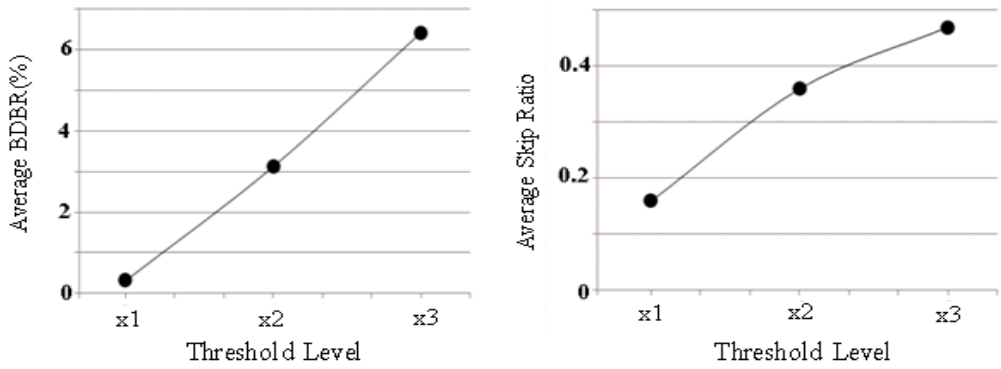


그림 3.7 Threshold Level 변화에 따른 평균 BD-BR increase 와 Skip Ratio

3.3.2 Simplified SSE & CABAC without Transform

앞서 제안된 Threshold algorithm은 DCT를 생략할 수 있는 방법을 보여주고 있으나, DCT가 생략되는 비율이 영상과 QP에 의해 달라지며, RD 저하를 최소화하는 영상 압축 연구 영역에서는 Threshold의 상한이 제한된다. 이 단락에서는 적응 처리 방식을 사용하지 않고, DCT를 생략할 수 있는 방법을 제한하고자 한다. 알고리즘의 제안을 위해, Threshold algorithm의 식(3-18)을 다시 보도록 한다. N_s는 그 정의에 의해, else 조건문의 N_s는 if 조건문의 N_s 보다 대체로 큰 값을 갖게 된다. 그러므로, else 조건문에서 Bits의 중요성은, if 조건문에서 보다 커지게 됨을 유추할 수 있다. 이와 같은 원리를 SSE에 적용하면, else

조건에서의 SSE의 중요성은 if 조건에서의 SSE보다 줄어들게 된다. 따라서, else 조건문에서의 SSE에 대해 보다 강한 단순화 기술을 적용한다 하더라도, RD 저하의 정도가 크게 심해지지 않을 수 있음을 유추할 수 있다. 식(3-16)의 SSE″는 Threshold(QP)가 증가함에 따라 빠르게 된다. 그러나, 실제 SSE는 quantization error이기 때문에, QP step의 크기에 의해 상한이 제한된다. 그러므로, SSE″이 증가한다 해도, 실제 SSE가 계속 증가하지 않는 경우가 발생하게 된다. 이 점을 고려하여, else 조건문의 SSE는 다음과 같이 Threshold(QP)로 표현된 SSE‴로 대체하기로 한다.

$$SSE''' = Threshold(QP) \times n \qquad (3-19)$$

이상의 모델링이 효과적인지를 판단하기 위해, 우선 아래와 같은 식(3-20)처럼 중간 단계의 알고리즘을 설정하고, 이 알고리즘에 의한 RD cost와 실제 RD cost를 비교하기로 한다. 중간 단계의 알고리즘이라 표현한 이유는 DCT를 하지 않으면 CABAC의 input에 해당하는 quantized transform coefficients가 존재하지 않기 때문에, 최종 알고리즘에서는 else 조건문의 Bits에 대해 다른 방법이 필요하게 된다.

$$\begin{aligned} &\text{if}(SSR_{avg} < Threshold(QP)) \qquad (3-20)\\ &\quad SSE = SSR;\ Bits = 0;\\ &\text{else}\\ &\quad SSE''' = Threshold(QP) \times n;\ Bits = \text{output of CABAC} \end{aligned}$$

아래 그림은 중간 단계의 알고리즘을 적용했을 때에, 실제 Real RD

cost를 x축을 하고 Estimated RD cost를 y축으로 하여, 32x32 DCT에 대한 실험 결과를 나타내고 있다. if 조건문에서는 Estimated RD cost가 Real RD cost에 비해 약간 큰 반면에, else 조건문에서는 Estimated RD cost가 Real RD cost에 비해 약간 작다. 그러나, 전체적으로 봤을 때, Estimated RD cost가 y=x인 추세선을 잘 따르고 있음을 관찰할 수 있다.

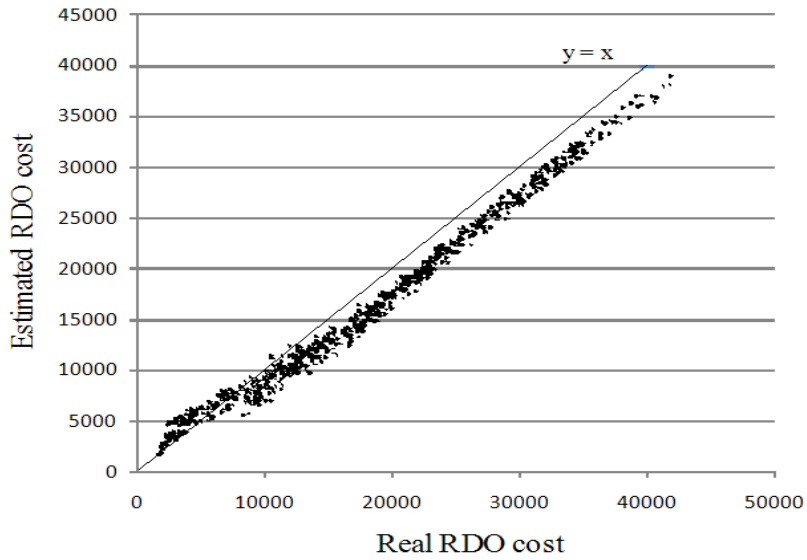


그림 3.8 식(3-20) 적용시, Real RD cost와 Estimated RD cost의 관계

아래 표 3.7은 식(3-20)에서 보여주는 중간 단계의 알고리즘을 적용했을 때의 BD-BR의 증가를 Threshold algorithm를 적용했을 때와 비교하여 보여주고 있다. 그 증가 정도가 Threshold algorithm을 적용했을 때에 비해 증가하였지만 여전히 미미한 수준임을 알 수 있다.

따라서, else 조건의 SSE를 식(3-19)로 대체하는 것이 RD 저하에 크게 영향을 주지 않음을 알 수 있다.

표 3.7 Threshold Algorithm와 식(3-20)의 알고리즘의 BD-BR의 증가 비교

Resolution	Threhold Method	(20)'s Algorithm
480p	0.27%	0.31%
1080p	0.35%	0.90%
Average	0.31%	0.61%

끝으로 else 부분의 Bits를 예측할 수 있는 방법을 제안하고자 한다. Bits는 앞서 제안된 Low-complexity CABAC에서 설명된 것처럼, N_s에 의해 예측될 수 있다. 따라서, N_s를 예측할 수 있으면, Bits도 예측이 가능하게 된다. Transform을 하지 않고, N_s 를 예측하기 위해, 잔차시호에 대한 SAD를 SAD_{res}로 표기하고, SAD_{res}와 N_s 관계를 분석하고자 한다. 확률적으로 SAD가 큰 잔차 신호는 SAD가 작은 잔차 신호보다 큰 값의 N_s를 갖게 된다. 이러한 유추는 실험에 의해서 확인하고 아래 그림 3.9에 그 결과를 나타내었다. x축은 SAD_{res}를 y축은 N_s 를 나타내고 있으며, 두 증가 관계가 정확한 일차 비례 관계를 나타내고 있지는 않지만, SAD_{res} 가 증가함에 따라서 N_s가 증가함을 확인할 수 있다. 그림의 "approximation"으로 표기된 선은 실험데이터와 추세선의 오류를 최소화도록, 기울기가 실험적으로 정해진 선이다.

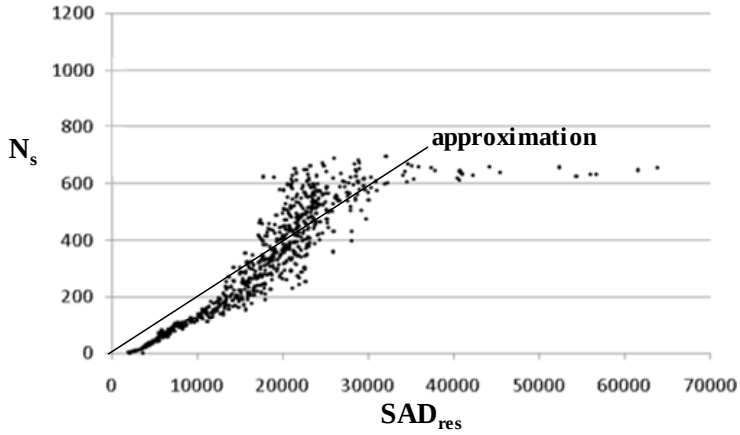


그림 3.9 N_s 와 SAD_{res}의 관계

이러한, 기울기는 QP와 DCT의 크기에 따라 달라지기 때문에, 기울기는 QP와 W_{TR} 에 종속되게 slope(QP, W_{TR}) 으로 표현된다. 이 기울기는 Class C의 모든 영상의 초반 20 frames에 대한 통계치로 결정되었다. 아래 표 3.8은 실제 실험에 사용된 slope(QP,W_{TR})를 나타내고 있다. 이상의 결과를 바탕으로, N_s 와 Bits 에 대한 예측식은 식(3-21)와 (3-22) 처럼 표현된다.

표 3.8 QP에 따른 slope(QP,W_{TR})

	W_{TR} 32	W_{TR} 16	W_{TR} 8	W_{TR} 4
QP22	0.044	0.044	0.042	0.042
QP27	0.022	0.022	0.020	0.030
QP32	0.013	0.013	0.014	0.017
QP37	0.008	0.008	0.007	0.009

$$N_s = \text{slope}\ (QP, W_{TR}) \times SAD_{res} \quad (3-21)$$

$$Bits = K(W_{TR}) \times \text{slope}\ (QP, W_{TR}) \times SAD_{res} \quad (3-22)$$

Threshold Algorithm에 식(3-19)와 식(3-22)를 적용한 최종 알고리즘은 식(3-24)처럼 정리될 수 있다. 본 논문에서는 이 알고리즘을 "Simplified SSE and CABAC without Transform"로 표기하도록 한다.

$$\begin{aligned} &\text{if}(SSR_{avg} < \text{Threshold}(QP, W_{TR})) \quad (3-24)\\ &\quad SSE = SSR;\ Bits = 0;\\ &\text{else}\\ &\quad SSE = \text{Threshold}(QP, W_{TR}) \times n;\\ &\quad Bits = K(W_{TR}) \times \text{slope}\ (QP, W_{TR}) \times SAD_{res};\end{aligned}$$

앞서 소개한 것처럼, HEVC에서는 다양한 크기의 DCT를 제공함으로써, "Simplified SSE and CABAC without Transform"를 DCT의 크기 별로 적용할 수 있다. 아래 표 3.9는 이상의 알고리즘을 DCT 크기 별로 누적해서 Inter mode와 Intra mode에 적용했을 때의 BD-BR의 증가를 보여주고 있다. 왼쪽의 칼럼은 생략되는 DCT를 의미한다. 예를 들어 Inter 32는 Inter mode에 32x32 DCT를 생략하고, 나머지 DCT에 대해서는 원래의 RDO 과정을 수행했음을 의미한다. "Inter/Intra all"은 모든 DCT를 생략했음을 의미한다. 모든 DCT에 적용했을 때의 BD-BR의 증가는 평균 13.14%로 상당히 큼을 알 수 있다. 따라서, 효과적으로 적용하기 위해서는 줄어드는 연산량 대비, BD-BR의 증가가

작은 DCT 크기에 선별적으로 적용해야 한다.

표 3.9 Simplified SSE and CABAC without Transform의 DCT별 적용시 BD-BR의 증가

	BD-BR Increase		
Skipped DCT	480p	1080p	Average
Inter 32	2.34%	2.68%	2.51%
Inter 32, 16	5.05%	6.75%	2.95%
Inter 32, 16, 8	8.73%	10.81%	9.77%
Inter all	8.74%	10.82%	9.78%
Inter all + Intra 32,16	10.65%	12.40%	11.53%
Inter all + Intra 32, 16, 8	11.58%	13.20%	12.39%
Inter all + Intra all	12.59%	13.68%	13.14%

3.4 Evaluation

이번 섹션에서는 이전 섹션까지 소개된 다양한 Simplified RDO 기술들의 조합과 그로 이한 RD 저하와 줄어든 연산량을 제시하고자 한다. 이전 섹션까지의 알고리즘 성능을 평가하기 위해 제시된 RD 저하는 Luma data에만 Simplified RDO를 적용했을 때의 결과이다. 이번 섹션에서는 Luma 뿐만 아니라 Chroma data에도 적용하여 BD-BR을 측정하였다. 줄어든 연산량을 제시하고자, encoding 수행시간을 측정하였고, 알고리즘을 적용시 적용하지 않은 encoding 수행시간에 비해 얼마나 줄어들었는지를 표시하였다. 이 실험을 위해, RDOQ는 on이고 low-delay configuration이 사용되었으며, Transform depth는 1로 설정되었다.

표 3.10 줄어든 연산량과 RD 저하 측정을 위한 실험의 Simplified RDO 기술 조합

Condition	Simplified SSE	Low-Complexity CABAC	Threshold Algorithm	32 TR	16TR	8TR	4TR
# 1	On	Off	Off	O	O	O	O
# 2	On	On	Off	O	O	O	O
# 3	On	On	Off	X	O	O	O
# 4	On	On	Off	X	O	O	X
# 5	On	On	Off	X	X	O	X
# 6	On	On	Off	X	X	X	X
# 7	On	On	On	O	O	O	O

위의 표 3.10은 줄어든 연산량과 RD 저하 측정을 위한 실험을 하기 위해 필요한 Simplified RDO 기술들의 조합을 8가지로 나누어서 보여주고 있다. Simplified SSE는 식(3-7)에 의해 SSE를 예측하는 것을 의미하고, Low-Complexity CABAC는 식(3-14)에 Bits를 예측하는 것을 의미한다. "X"에 의미는 해당하는 transform을 생략하겠다는 의미이고, "O"는 생략하지 않겠다는 의미이다. 32x32 transform의 생략이 다른 transform에 비해 연산량을 가장 많이 줄이며, 줄어든 연산량 대비 RD 저하는 제일 작은 편이다. 그래서, Condition 3에서 32x32 transform을 선택하였고, 그 다음 Condition 4에서 4x4 transform을 선택한 것도 비슷한 이유에서이다. 상대적으로 16x16 과 8x8 transform의 생략이 줄어든 연산량 대비 RD 저하가 큰 편이다. Condition 7은 Threshold Algorithm을 적용했을 경우이다.

아래 그림 3.10은 표 3.10을 반영한 실험 결과를 나타내는 것으로, x축은 encoding 수행시간을 y축은 BD-BR 증가를 나타낸다. Condition 5, 6 에 비해 Condition 1, 2, 3, 4가 상대적으로 BD-BR의 증가가 작으면서 수행시간이 크게 감소하는 모습을 보인다. 따라서, Condition 3 또는 4 가 연산량을 크게 줄이면서 RD 저하를 최소화하는 선택임을 알 수 있다. Condition 3에서 BD-BR의 증가는 5.13%이고 수행시간은 7.1% 감소하고, Condition 4에서 BD-BR의 증가는 5.93%이고 수행시간은 7.8% 감소한다.

그림 3.11은 Condition 4에 세가지 복잡도의 CABAC를 적용했을 때의 결과이다. 그림의 L,M,H 는 각각 High, Medium, Low complexity

CABAC을 표시한다. 복잡도가 커질수록 BD-BR이 감소하나, 감소되는 수행시간도 함께 줄어는 경향을 보인다. 그러므로, 목표하는 연산량의 감소나 RD 저하를 기준으로 특정 복잡도의 CABAC을 선택하면 된다.

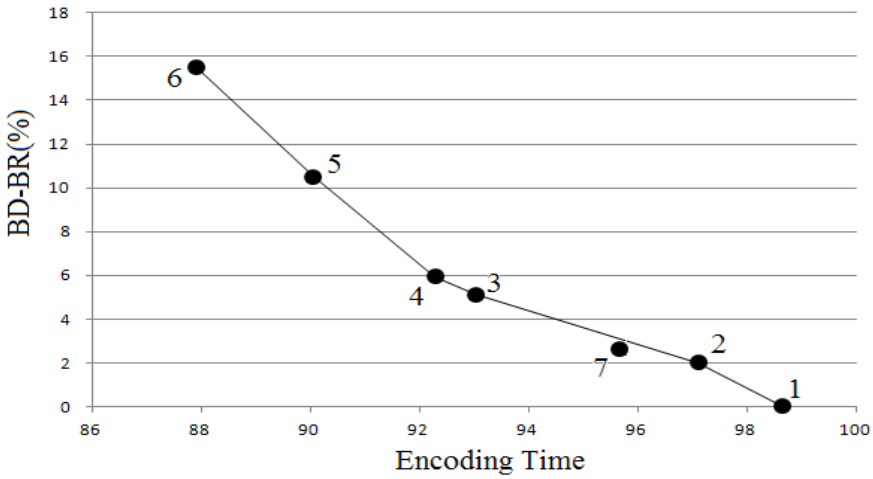


그림 3.10 다양한 Simplified RDO 기술 조합에 따른 수행시간과 BD-BR의 증가

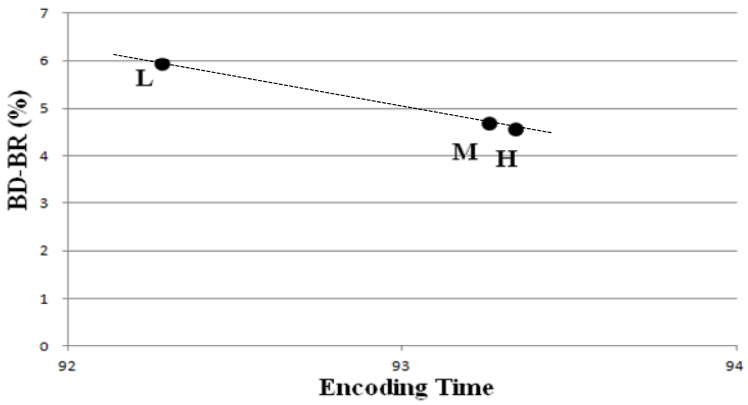


그림 3.11 Condition 4에 세가지 복잡도 CABAC을 적용한 실험결과

제 4 장 Zero Block Detection

4.1 Extension of H.264/AVC Zero Block Detection for HEVC

이 섹션에서는 HEVC에서의 ZB의 특징과 결정 과정을 분석하고자 한다. 그리고 나서, H.264/AVC에 대해 ZB detection을 제안한 연구[20]를 확장하여 HEVC에 적용해보도록 하겠다.

4.1.1 Characteristics of the zero blocks in HEVC

HEVC에서는 기본적으로 모든 quantized transform coefficients가 0일 때, 해당 블록은 ZB으로 encoding된다. 또한, non-zero quantized transform coefficients가 존재하더라도, 해당 블록을 ZB로 가정했을 때의 RD cost가 non-zero quantized transform coefficients을 기반으로 계산된 RD cost보다 작으면 ZB으로 encoding된다.

아래 그림 4.1은 HEVC에서의 ZB 결정 과정을 보여주고 있다. 첫 번째로는 입력 블록의 모든 quantized transform coefficients가 0인지를 확인하고, 만약 참이면 ZB으로 결정하고, 이 ZB은 GZB에 해당하게 된다. 만약 거짓인 경우, RD cost를 비교를 수행하게 된다. $Cost_{\text{nonnull}}$ 은

DCT/Q의 결과를 기반으로 연산된 RD cost를 의미한다. 이와 동시에, 모든 quantized transform coefficients를 강제로 0으로 하여 bitstream을 발생시키지 않도록 할 수 있다. 이러한 강제 상황을 기반으로 계산된 RD cost를 $Cost_{\text{null}}$ 로 표기한다. 이런 강제의 결과로 부정확한 coefficients이 발생하여 distortion이 크게 증가할 수 있게 된다. 그러나, 이러한 강제는 quantized transform coefficients에 대한 bitstream를 발생시키지 않기 때문에, 결과적으로 RD cost를 작게 만들 수도 있다. 따라서, 해당 블록이 non-zero quantized transform coefficients을 갖는다고 해도 아래 식(4-1)을 만족하면, RD 성능이 개선된 결과를 제공할 수 있게 된다.

$$Cost_{\text{null}} < Cost_{\text{nonnull}} \qquad (4-1)$$

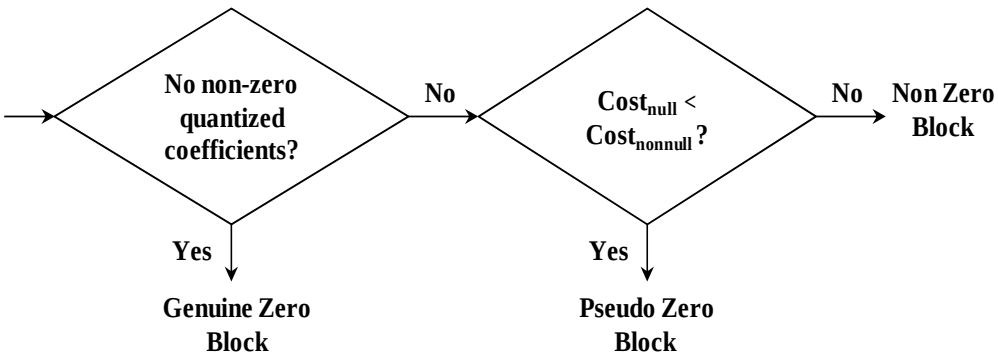


그림 4.1 HEVC에서의 ZB 결정 과정

이렇게 식(4-1)을 만족하는 블록은 그림 4.1에서 두 번째 조건문을

만족하는 ZB으로 PZB에 해당하게 된다.

H.264/AVC의 작은 크기의 블록에 대해서 대부분의 ZB이 GZB이기 때문에, 이러한 GZB와 PZB의 구분이 중요하지 않다. 그러나, HEVC의 큰 크기의 블록에 대해서 PZB의 출현빈도가 크게 증가하게 된다. 이러한 이유로, PZB에 대한 검출은 모든 ZB의 정확한 검출을 위해 꼭 필요한 요소임을 알 수 있다. 아래 표 4.1은 전체 ZB 대비 PZB의 비율을 제시하고 있다. 이 표를 통해, 블록이 클수록 그리고 QP가 작을수록 PZB의 비율이 증가함을 알 수 있다. 특히, 16x16 과 32x32 의 경우 PZB의 비율이 상당하기 때문에, PZB 검출 역시 GZB 검출 못지 않게 중요함을 알 수 있다. 4x4 와 8x8에 대한 PZB의 비율은 상대적으로 작지만, 이는 전체 ZB의 개수를 증가시키는 데에 영향을 주고 있다.

표 4.1 전체 ZB 대비 PZB의 비율

Sequence	DCT	QP22	QP27	QP32	QP37
BQMall (480P)	4×4	6.1%	4.4%	3.0%	1.8%
	8×8	13.2%	9.8%	6.8%	4.2%
	16×16	22.4%	17.4%	13.4%	10.4%
	32×32	30.9%	27.0%	21.5%	18.7%
BQTerrace (1080P)	4×4	13.3%	6.2%	2.0%	0.7%
	8×8	40.1%	20.9%	7.2%	2.4%
	16×16	66.4%	41.9%	18.6%	7.4%
	32×32	90.9%	74.6%	41.8%	18.6%

4.1.2 ZB detection by an extension of the H.264/AVC algorithm

이 섹션에서는 연구[4]에서 제안된 ZB detection 알고리즘을 간략히 소개하고, HEVC에 적용하기 위해 필요한 과정을 소개하도록 한다. 설명을 간략하기 위해, 4x4 블록에 대해서만 다루고, 설명된 알고리즘은 같은 방식으로 8x8, 16x16, 32x32 에 대해서 확장 적용한다. 연구[20]에서, Hadamard transform의 결과가 ZB detection에 활용되는데, 그 이유는 4x4 DCT의 basis function과 4x4 Hadamard transform의 basis function이 매우 유사하기 때문이다.

$$Y = CXC^T \quad \leftrightarrow \quad Z = HXH^T \qquad (4-2)$$

식(4-2)에서, 윗첨자 T는 해당 행렬의 transpose 를 표기하고, X는 4x4 잔차신호를, 행렬 C와 H는 각각 DCT와 Hadamard transform 행렬을 의미한다. H.264/AVC의 해당 행렬의 목록은 다음과 식(4-3)과 같다.

$$\mathbf{C} = \begin{pmatrix} C0 \\ C1 \\ C2 \\ C3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix} \quad \mathbf{H} = \begin{pmatrix} H0 \\ H1 \\ H2 \\ H3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \qquad (4-3)$$

H의 basis function H0와 H2는 C의 basis function C0와 C2와 동일하며, 나머지 basis function들도 비슷하다고 할 수 있다. 따라서,

y[i][j] 의 값은 z[i][j]에 값과 유사하게 된다. 이 알고리즘에서는 $TH_{DCT}[QP][i][j]$ 을 ZB detection을 위한 주어진 위치(i,j)와 QP에 대한 threshold로 정의한다. 즉, 모든 (i,j)에 대해서 y[i][j]가 $TH_{DCT}[QP][i][j]$보다 작으면 ZB이 되어야 한다. 연구[20]에서는 $TH_{H}[QP][i][j]$ 는 Hadamard transform에 대한 threshold로 표기하고 $TH_{DCT}[QP][0][0]$ 로 설정한다.

$$TH_{H}[QP][i][j] = TH_{DCT}[QP][0][0] \quad i,j \in [0,3] \qquad (4-4)$$

연구[20]는 Hadamard coefficient인 z[i][j] 가 $TH_{H}[QP][i][j]$보다 작으면 이에 대응되는 quantized DCT coefficient가 0이 될 것이라고 가정한다. 따라서, 모든 Hadamard coefficient가 $TH_{H}[QP][i][j]$ 보다 작으면, 해당하는 4x4 블록은 DCT/Q 과정 이전에 ZB으로 검출할 수 있다.

연구[20]의 HEVC에 대한 적용은 간단하다. HEVC의 basis function이 scaling 요소를 배제하면 H.264/AVC의 basis function에 비해 약간 달라졌으나 거의 비슷하고, 여전히 DCT의 C0와 C2는 Hadamard transform의 H0와 H2와 같다. 그러므로, 연구[20]에서 제안된 threshold 방법은 HEVC에서도 적용이 가능하다. 그러나, HEVC의 quantization 과정이 약간 다르기 때문에 식(4-4)에서 표현된 threshold의 값을 HEVC에 적용하기 위해서는 약간의 수정 과정이 필요하다. 아래 식(4-5)와 식(4-6)은 HEVC의 quantization process를

설명하고 있고, 식(4-7), 식(4-8) 그리고 식(4-9)는 HEVC에 적합한 threshold 값을 끌어내는 과정을 설명하고 있다. HEVC의 quantization 과정은 식(4-5)에 의해 표현되고 있다.

$$QC = (C \times mult + offset) / 2^{\wedge}shift \qquad (4-5)$$

이 식에서 QC 와 C는 각각 quantized coefficient와 DCT coefficient를 의미한다. 또한, HEVC에서의 *mult, offset* 그리고 *shift* 다음과 같이 식(4-6)처럼 정의된다.

$$\begin{aligned} shift &= \text{Q\_SHIFT} + \text{D\_RANGE} + iPer - BitDepth - w \qquad (4-6)\\ offset &= 85 \times 2^{\wedge}(shift-9) \quad \text{for non-I slice} \\ &\quad\ 171 \times 2^{\wedge}(shift-9) \text{ for I slice} \\ mult &= \text{Q\_Array}[iRem] \end{aligned}$$

식(4-6)에서, 각 파라미터들은 다음과 같이 설정된다. Q_SHIFT=14, D_RANGE=15, $iPer = QP/6$, $w = log_2((DCT\ width))$, $iRem = QP\%6$ *and* Q_Array=*{26214, 23302, 20560, 18396, 16384, 14564}*. *BitDepth* 는 pixel을 표현하는 bit의 크기를 나타내는 것인데, HEVC에서는 기본적으로 8로 설정된다. *offset* 의 값은 *shift* 와 주어진 slice 타입에 의해 결정된다. 거의 대부분의 ZB는 non-I slice에 존재하기 때문에, 본 논문에서는 non-I slice의 *offset* 만 다루도록 한다. H.264/AVC에서는 ***mult*** 의 값이 coefficient의 위치에 따라 다른 값을 같지만, HEVC에서는 coefficient의 위치에 상관없이 ***mult*** 의 값이 같다.

따라서, QC는 오직 QP와 W_{DCT} (DCT의 크기)에만 영향을 받게 된다. 임의의 ZB은 모든 quantized transform coefficients가 0이기 때문에, 이는 식(4-7)과 같은 부등식을 만족하게 된다.

$$QC = (C \times mult + offset) / 2^{shift} < 1 \quad (4-7)$$

식(4-7)의 좌변과 우변을 적적하게 재배치하면 식(4-8)과 같이 된다.

$$C < (2^{shift} - offset)/mult \quad (4-8)$$

그러므로, HEVC에 적용될 새로운 threshold의 값은 $TH_{DCT}[QP][W_{DCT}]$로 표기되며, 식(4-9) 와 같이 표현된다.

$$TH_{DCT}[QP][W_{DCT}] = (2^{shift} - offset)/mult \quad (4-9)$$

그 다음 과정은 식(4-9)로부터 Hadamard coefficient에 적용할 threshold 값을 유도하는 것이다. 기본적으로는 식(4-4)와 같지만, HEVC에서는 DCT와 Hadamard transform 이 unitary matrix 조건을 만족하기 위한 normalization factor가 다르기 때문에 이를 보정하는 값이 필요하다. 식(4-10)에서의 m은 보정하는 값을 의미하며, m의 값은 DCT의 크기에 따라 다르다. DCT 32x32, 16x16, 8x8, 4x4에 해당하는 m의 값은 순서대로 8, 2, 1/2, 1/8이다.

$$TH_H[QP][W_{DCT}] = TH_{DCT}[QP][W_{DCT}] \times m \quad (4-10)$$

이상의 threshold를 적용한 ZB detection 알고리즘은 다음 식(4-11)과 같은 조건문으로 표현된다.

$$\text{if}(NUM_{hadamard} = 0) \quad ZB = 1 \qquad (4-11)$$

식(4-11)에서 $NUM_{hadamard}$ 는 식(4-10)에서의 threshold 보다 큰 Hadamard coefficients의 개수를 의미하며, "ZB=1"은 해당 블록이 ZB으로 검출되었다는 것을 가리킨다. 식(4-11)로 정리된 알고리즘은 H.264/AVC 알고리즘의 HEVC용 확장 버전으로 본 논문에서는 새로 제안하는 ZB detection 알고리즘의 성능을 평가하기 위한 reference로 사용한다. H.264/AVC에서는 4x4 와 8x8 크기의 transform 블록을 지원하지만, 16x16 이나 32x32 크기의 transform 블록을 지원하지는 않는다. 따라서, HEVC에서도 기존의 알고리즘이 4x4 와 8x8 크기의 블록에서는 효과적일 것이라고 예상이 가능하지만, 16x16이나 32x32 크기의 블록에서는 기존의 알고리즘이 효과적일지는 분명하지가 않다. 따라서, 큰 블록에 대한 효과적인 ZB detection을 위한 추가적인 조건이 다음 섹션에서 논의될 것이다.

4.2 Zero Block Detection for HEVC

이번 섹션에서는 HEVC에 적합한 새로운 ZB detection 알고리즘을 제안하고자 한다. ZB detection의 목적이 RDO 연산을 줄이는 것이기 때문에, 제안된 알고리즘을 수행하기 위한 연산량 또한 중요한 이슈이다. 따라서, 제안된 알고리즘은 높은 ZB 검출율을 유지하면서, 알고리즘의 수행을 위한 연산의 복잡도를 낮게 유지하는 것을 목표로 한다. GZB와 PZB를 검출하기 위한 조건을 제안하고, ZB detection의 연산의 복잡도를 줄이는 방법을 제시하며, 최종적으로 이러한 제안을 종합하여 HEVC에 맞는 새로운 ZB detection 알고리즘을 제안한다.

4.2.1 GZB detection for 16x16 and 32x32 transforms

이번 서브 섹션에서는 16x16과 32x32 크기의 블록에 대한 ZB detection을 분석하도록 한다. 기존 알고리즘의 HEVC용 확장 버전은 16x16과 32x32 블록 크기에 대한 Hadamard transform 수행해야 한다. 그러나, 큰 블록에 대한 Hadamard transform의 연산의 복잡도는 4x4나 8x8에 비해 매우 크고, FME 과정에서는 해당 크기의 Hadamard transform 을 수행하지 않기 때문에, 이를 항상 추가로 수행해야 하기 때문에, 연산량의 증가를 초래하게 된다. 앞서 설명되었던 것처럼, 4x4 또는 8x8 크기의 Hadamard transform이 FME 과정에서 이미 수행되기 때문에, 해당 크기의 블록에 대해서는 추가 연산량이 필요 없다. 이와

비교하면, 16x16과 32x32 블록 크기에 대한 Hadamard transform 수행으로 인한 상대적인 연산량의 증가가 매우 크기 때문에, ZB detection의 연구 목적인 RDO 연산량의 효율적인 감소에 적합하지 않음을 알 수 있다. 따라서, 16x16과 32x32 블록 크기에 대한 Hadamard transform 수행을 피하면서도, 특정 주파수에 에너지가 쏠리지 않는지의 여부를 확인할 수 있는 방법이 제안되어야 효과적으로 RDO의 연산량을 감소시킬 수가 있다.

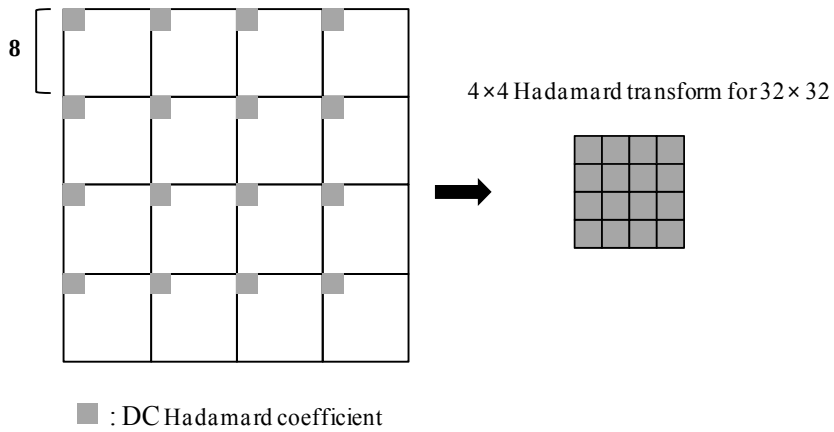


그림 4.2 32x32 block의 ZB detection을 위한 8x8 Hadamard DC coefficients로 이루어진 4x4 Hadamard trasnform

32x32 Hadamard coefficients가 특정 주파수 성분으로 쏠리지 않는지를 예측하기 위해, 16개의 8x8 Hadamard transform의 DC coefficients를 활용하기로 한다. 그림 4.2는 32x32가 8x8 크기의 4x4 블럭으로 분할되는 모습을 보여주고 있다. 각각의 8x8 왼쪽 위 모서리에 해당하는 부분이 해당 8x8 Hadamard transform의 DC coefficient를

나타내며, 이런 DC coefficients만을 대상으로 4x4 블록을 구성한다. 이렇게 구성된 4x4 블록에 대해서, Hadamard transform을 수행하면, 그림의 오른쪽의 4x4 Hadamard coefficients로 구성된다. 본 논문에서는, 구분을 위해 이를 DC Hadamard transform으로 표기하도록 한다. 8x8 Hadamard transform의 DC coefficient는 8x8 잔차 신호의 합을 의미하기 때문에, DC Hadamard transform은 32x32 Hadamard transform의 저주파 성분 4x4를 제공하는 것이고, 결국 32x32 DCT의 저주파 성분의 4x4에 대한 단순화된 버전을 제공한다고 볼 수 있다. 최종적으로, DC Hadamard transform에 의해 계산된 coefficients들이 식(4-10)에서 32x32 크기일 때의 $TH_H[QP][W_{DCT}]$보다 작은 지를 테스트하여, 해당 32x32 블록이 ZB 인지를 검출할 수가 있다. DC Hadamard transform에 의한 추가되는 연산량이 크지 않기 때문에, 효과적으로 RDO의 연산량을 줄일 수 있다. 16x16의 경우는 8x8 크기의 2x2로 분할되고, 똑같은 방식으로 2x2 DC coefficient를 계산하여, $TH_H[QP][W_{DCT}]$와 비교를 통해 해당 16x16 블록이 ZB 인지를 검출할 수가 있다.

이상 위에서 설명된 16x16과 32x32 블록에 대한 ZB detection 알고리즘과 기존의 알고리즘을 묶어서 정리하면 식(4-12)같은 조건문이 된다.

$$\text{if}(NUM_{hadamard} = 0 \;\&\&\; DC_{hadamard} = 0) \quad GZB = 1 \tag{4-12}$$

앞서 설명된 것처럼, $NUM_{hadamard}$ 는 식(4-10)에서의 threshold 보다 큰 Hadamard coefficient의 개수를 의미한다. 혼동을 피하기 위해, 좀 더 상세히 설명하자면, 여기서 언급된 Hadamard coefficient는 16x16 이나 32x32 Hadamard transform에 의한 coefficient가 아니라, 분할된 8x8 Hadamard transform 통해 계산된 coefficient를 의미한다. 반면, $DC_{hadamard}$는 식(4-10)에서의 threshold 보다 큰 DC Hadamard coefficient의 개수를 의미한다. $DC_{hadamard}$ 는 16x16 또는 32x32의 저주파 성분인 2x2 또는 4x4에 대응되기 때문에, "$DC_{hadamard}=0$"로 구성된 추가 조건은 16x16이나 32x32 Hadamard transform의 저주파 성분인 2x2 또는 4x4가 식(4-10)의 threshold 조건을 만족함을 보장할 수가 있게 된다.

4.2.2 Relaxed conditions for PZB detection

식(4-12)는 기존 알고리즘을 기반으로 연산의 복잡도를 낮춘 ZB detection 알고리즘을 제시하고 있다. 그러나, 이 알고리즘은 GZB만을 검출하도록 설계된 알고리즘이기 때문에, 많은 수의 PZB의 검출을 놓칠 수 밖에 없다. 이를 개선하기 위해서, 이 서브 섹션에서는 식(4-10)에서의 threshold보다 큰 Hadamard coefficient를 갖는 PZB을 검출하기 위해, 식(4-12)의 조건을 수정하였다. PZB을 검출하기 위한 새로운 threshold를 정하기 위해서, threshold의 값을 변화시키면서 PZB에 검출의 성능을 측정하는 실험이 진행되었다. 새로운 threshold는

잘못된 결정으로 인한 RD 저하를 최소화하면서도 최대한 RDO 연산량을 줄일 수 있는 값으로 정해졌다. (이에 대한 실험은 다음 섹션의 그림 4.7에서 확인할 수 있다.) 이 새로운 threshold는 식(4-10)에서의 threshold의 2배의 값으로 정해졌고, 이는 식(4-13)과 같이 정리된다.

$$TH_{H\_PZB}[QP][W_{DCT}] = 2 \times TH_{H}[QP][W_{DCT}] \quad (4-13)$$

식(4-13)에서 $TH_{H_PZB}[QP][W_{DCT}]$는 PZB 검출을 위한 새로운 threshold값을 나타낸다. 이러한 새로운 threshold를 그대로 식(4-12)에 적용하면, 식(4-14)와 같은 조건문이 구성된다.

$$\text{if}(NUM_{hadamard\_x2} = 0 \;\&\&\; DC_{hadamard} = 0) \quad (4-14)$$
$$PZB = 1$$

$NUM_{hadamard_x2}$는 식(4-13)의 threshold 보다 큰 Hadamard coefficients의 개수를 의미한다. 이를 조건을 그대로 수행하면, 많은 PZB가 검출될 수 있으나, Non-ZB을 PZB으로 결정하는 false decision의 비율이 높아지는 경향이 있어서, 식(4-14)의 두 번째 조건을 좀 더 제한하도록 하여, 식(4-15)와 같은 조건문을 구성하였다.

$$\text{if}(NUM_{hadamard\_x2} = 0 \;\&\&\; NUM_{4x4} = 0) \quad (4-15)$$
$$PZB = 1$$

NUM_{4x4} 는 식(4-10)의 threshold보다 큰 16x16 (또는 32x32)의 저주파 성분 4x4 Hadamard coefficient의 개수이다. 16x16에 대해서는

분할된 8x8 Hadamard transform의 결과로 저주파 성분 4x4 Hadamard coefficient를 계산할 수 있다. 그래서, 16x16 Hadamard transform을 추가로 수행해야 되는 연산의 증가를 회피할 수 있다. 또한, 32x32 블록에 대해서는 NUM_{4x4} 가 4x4 DC Hadamard transform와 동일하기 때문에, 추가로 요구되는 연산량은 없다.

표 4.2 (4-12)와 (4-15)의해 결정되는 PZB와 GZB의 상대적 비율

			PZB		GZB	
Sequence	QP	DCT	(12)	(15)	(12)	(15)
BQMall (480p)	22	16x16	17%	83%	78%	22%
		32x32	11%	89%	64%	37%
	27	16x16	20%	80%	86%	14%
		32x32	18%	82%	72%	28%
	32	16x16	22%	78%	91%	9%
		32x32	19%	82%	82%	18%
	37	16x16	24%	76%	94%	6%
		32x32	22%	78%	87%	13%
BQTerrace (1080p)	22	16x16	19%	81%	85%	16%
		32x32	22%	78%	81%	19%
	27	16x16	18%	82%	88%	12%
		32x32	20%	80%	87%	13%
	32	16x16	28%	72%	92%	8%
		32x32	27%	73%	84%	16%
	37	16x16	40%	60%	96%	4%
		32x32	35%	65%	90%	10%

식(4-12)와 식(4-15)에 의한 ZB detection의 효과를 확인하기 위해, 표 4.2에서는 ZB 검출비율이 제시되고 있다. PZB 와 GZB 열은 각각 PZB와 GZB의 상대적인 식(4-12)와 식(4-15)에 의한 상대적인 검출 비율을 나타내고, '(12)' 와 '(15)' 는 각각 식(4-12)와 식(4-

15)의 조건문을 의미한다. 표 4.2에서 확인할 수 있듯이, 식(4-12)의 조건이 효과적으로 GZB를 검출하는데 반해, PZB을 검출하지 못하고 있다. 이렇게 검출되지 못한 PZB는 식(4-15)에 의해 검출되게 된다.

4.2.3 Further complexity reduction with SAD (or SATD) test

HEVC에서는 SAD (or SATD)의 값은 prediction 과정에서 연산된다. 따라서 RDO 전에 미리 연산된 값이 있을 때는 추가 연산 없이, SAD (or SATD)의 값을 얻을 수 있다. 이번 서브 섹션에서는, 이 값을 활용하여, 해당 블록이 ZB인지 아닌 지를 구분할 수 있는 방법을 설명하고자 한다. 이런 SAD (or SATD)을 활용하면 ZB detection의 연산의 복잡도를 효과적으로 줄일 수 있다.

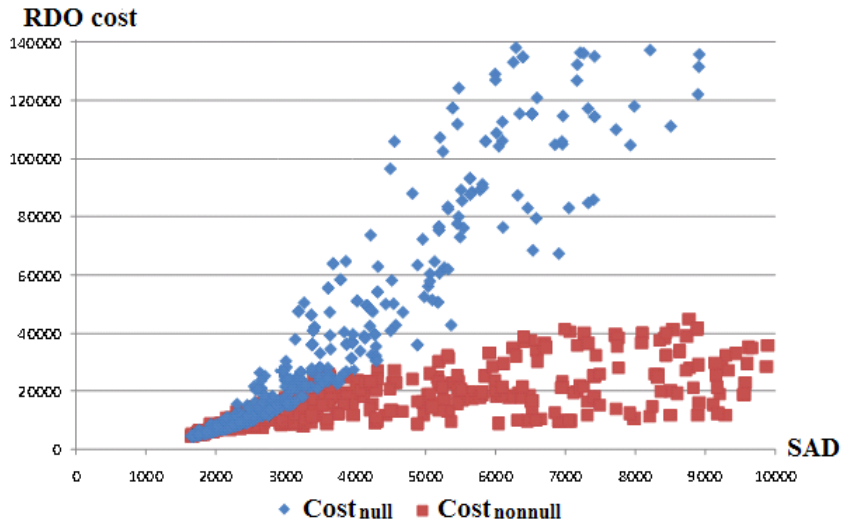


그림 4.3 32x32 block에 대한 SAD대비 $Cost_{null}$과 $Cost_{nonnull}$의 분포

이를 위해, HEVC에서 PZB의 결정 과정을 다시 보도록 한다. 그림 4.1에 나타난 것처럼 PZB는 $Cost_{null}$ 과 $Cost_{nonnull}$ 의 비교를 통해 결정된다. 그림 4.3은 32x32 DCT에 대한 $Cost_{null}$ 과 $Cost_{nonnull}$ 의 값의 분포를 보여주고 있다. 이 때 x축은 잔차신호의 SAD를 가리킨다. $Cost_{null}$ 은 SAD가 증가함에 따라 매우 급격하게 증가하는 반면에, $Cost_{nonnull}$ 의 경우는 SAD 증가함에 따라, $Cost_{nonnull}$의 증가의 폭이 줄어듬을 알 수 있다. 그러므로, SAD 값이 특정값 보다 클 경우에는 $Cost_{null}$이 $Cost_{nonnull}$ 보다 항상 커지게 되기 때문에, 해당 블록은 ZB이 아닌 것으로 예측할 수 있다. 따라서, SAD가 이 특정값보다 작다는 것이 ZB detection을 위한 새로운 조건이 될 수 있고, 이 값은 QP와 DCT의 크기에 영향을 받기 때문에, $UB_{SAD}[QP][W_{DCT}]$로 표기하도록 한다. SAD가 증가함에 따라, $Cost_{null}$과 $Cost_{nonnull}$ 의 차이 또한 빠르게 커진다. 따라서, 상대적으로 큰 SAD의 값을 지닌 블록이 잘못 ZB으로 결정될 경우 RD 저하가 더 증가하게 된다. 따라서, $UB_{SAD}[QP][W_{DCT}]$을 정확하게 정하는 것은 작은 RD 저하를 목표로 하는 ZB detection 에 중요한 요소임을 알 수 있다.

$UB_{SAD}[QP][W_{DCT}]$를 정하기 위해, HEVC에서 사용되는 RD cost를 분석하도록 한다. RD cost는 원신호와 재구성 신호를 기반으로 계산된 SSE로 대표되는 "Distortion" 부분과 발생되는 bitstream의 bits량으로 대표되는 "Rate"부분으로 이루어져 있다.

$$Cost_{\text{null}} = SSE_{\text{null}} + \lambda \times Bits_{\text{null}} \quad (4-16)$$

$$Cost_{\text{nonnull}} = SSE_{\text{nonnull}} + \lambda \times Bits_{\text{nonull}} \quad (4-17)$$

식(4-16)에서 SSE_{null} 과 $Bits_{\text{null}}$ 은 encoding되는 residual data가 없는 블록의 distortion과 bits를 나타내고 있고, 반면에, SSE_{nonnull} 과 $Bits_{\text{nonull}}$ 은 encoding되는 residual data가 있는 블록의 distortion과 bits를 나타낸다. $Bits_{\text{null}}$ 은 zero CBF에 대한 bits가 반영되는 것으로 그 값이 SSE_{null} 에 비해 매우 작기 때문에, 간편한 유도를 위해 $Cost_{\text{null}}$ 를 다음과 같이 근사화 한다.

$$Cost_{\text{null}} \approx SSE_{\text{null}} \quad (4-18)$$

$Cost_{\text{null}}$ 과 $Cost_{\text{nonnull}}$ 로부터 $UB_{SAD}[QP][W_{DCT}]$를 끌어내기 위해, 다음과 같은 신호들을 정의하도록 한다.

S: original signals

RS: reconstructed signals

P: predicted signals

R: residual signals

RR: reconstructed residual signals

SSE를 구성하는 R과 RS는 다음 식(4-19)처럼 R과 RR로 표현이 가능하다.

$$SSE = \| S - RS \|^2 \quad (4-19)$$
$$= \| (P+R) - (P+RR) \|^2 = \| R-RR \|^2$$

$Cost_{null}$은 encoding 되는 residual data가 없기 때문에, RR의 모든 원소들이 0이 됨을 의미한다. 따라서, 식(4-18)에서의 $Cost_{null}$ 은 다음 식(4-20)과 같이 표현된다.

$$Cost_{null} = SSE_{null} = \| R \| = R_1^2 + \cdots\cdots + R_n^2 \quad (4-20)$$

Inter prediction에서 발생하는 대부분의 잔차신호들은 그 크기가 매우 작다. 만약, 잔차신호의 크기가 모두 0과 1로만 이루어졌다면, 식(4-20)의 $Cost_{null}$ 은 다음 식(4-21)과 같이 표현이 가능하다.

$$Cost_{null} = (|R_1| + \cdots + |R_n|) \times (|R_1| + \cdots + |R_n|) = SAD^2 \quad (4-21)$$

그러나, 실제로는 1보다 큰 크기를 갖는 잔차 신호들이 존재하기 때문에, 식(4-21)관계가 항상 성립할 수는 없다. 따라서, 이러한 잔차 신호들을 보정하기 위한 $C[W_{DCT}]$ 를 설정하여, 다음 식(4-22)와 같이 SAD로 표현되는 식을 구성할 수 있다.

$$Cost_{null} = SAD^2 / C[W_{DCT}] \quad (4-22)$$

$C[W_{DCT}]$는 DCT 크기 즉, W_{DCT} 에 따라 결정되는 값으로, 실험적으로 구해졌다. 32x32,16x16,8x8,4x4 DCT에 대하여, $C[W_{DCT}]$는 순서대로,

580, 148, 37, 17 로 설정되었다. $C[W_{DCT}]$ 의 값의 변화에 따른 ZB detection 성능의 sensitivity는 이후에 실험 결과와 함께 제시하도록 한다. 결과적으로, 식(4-22)는 $Cost_{null}$ 에 대한 근사값을 SAD로만 계산되도록 하는 근사식을 제시하고 있다.

그 다음의 과정으로는 식(4-17)의 $Cost_{nonnull}$ 를 미리 알고 있는 값들로 구성될 수 있게 유도하는 것이다. DCT가 unitary matrix라고 가정하면 (아래식의 T는 DCT matrix를 표기함), $Cost_{nonnull}$ 의 SSE는 다음 식(4-23)과 같이 표현된다.

$$\begin{aligned} SSE &= \| T(R - RR)T^T \|^2 \\ &= \| T(R)T^T - T(RR)T^T \|^2 \\ &= \| F - F' \|^2 \end{aligned} \qquad (4-23)$$

식(4-23)에서 와 F와 F'은 각각 forward 그리고 backward transform을 나타낸다. HEVC의 DCT가 수학적으로 완벽한 unitary matrix는 아니지만 거의 같기 때문에, 식(4-23)의 SSE에 대한 근사는 합리적인 근사로 볼 수 있다. 식(4-23)에서 실제 DCT의 matrix가 적용되면 unitary matrix 가 되기 위한 normalization factor가 고려되어 계산되어야 하는데, 설명의 간략화를 위해, normalization factor를 명시적으로 표기하지 않았다.

대체로 PZB의 $Cost_{nonnull}$은 bits의 량이 상대적으로 작기 때문에, bits에 해당하는 부분을 0으로 가정하였다. 이러한 가정을 기반으로, $Cost_{nonnull}$은 다음과 같이 표현된다.

$$\mathrm{Cost}_{\mathrm{nonnull}} = \| F - F \|^2 \qquad (4-24)$$
$$= | F_1 - F_1' |^2 + \ldots\ldots\ldots + | F_n - F_n' |^2$$

$|F_1-F_1'|$은 quantization error를 의미하고, 이 값은 식(4-10)에서의 threshold 보다 작다. 이러한 이유로, $Cost_{\mathrm{nonnull}}$ 의 증가가 제한이 된다.

$$| F_i - F_i' | < \mathrm{TH}_{\mathrm{DCT}}[\mathrm{QP}][\mathrm{W}_{\mathrm{DCT}}] \qquad i \in [1, \mathrm{W}_{\mathrm{DCT}}^2] \qquad (4-25)$$

$Cost_{\mathrm{nonnull}}$ 대한 upper bound를 구하기 위해, 초반 20 frame의 480p 영상을 통해 $|F_i - F_i|$의 분포를 실험적으로 구하였다.

$\mathrm{UB}_{|\mathrm{F-F'}|}[\mathrm{QP}][\mathrm{W}_{\mathrm{DCT}}]$ 는 $|F_i - F_i'|$ 분포의 95%를 가리키는 값으로, 이를 활용하면, 대부분의 경우에 대해서 아래와 같은 부등식이 성립함을 알 수 있다.

$$Cost_{\mathrm{nonnull}} < \sum (\mathrm{UB}_{|\mathrm{F-F'}|}[\mathrm{QP}][\mathrm{W}_{\mathrm{DCT}}])^2 \qquad (4-26)$$

식(4-22)와 식(4-26)을 비교하게 되면, $Cost_{\mathrm{null}} < Cost_{\mathrm{nonnull}}$ 은 PZB를 검출하는 조건이 되고, 이는 아래와 같이 표현된다.

$$\mathrm{SAD}^2/\mathrm{C}[\mathrm{W}_{\mathrm{DCT}}] < \sum(\mathrm{UB}_{|\mathrm{F-F'}|}[\mathrm{QP}][\mathrm{W}_{\mathrm{DCT}}])^2 \qquad (4-27)$$

식(4-27)으로부터, SAD에 대한 조건을 유도하기 위해, 부등식을 SAD를 중심으로 정리하면 다음과 같다.

$$\mathrm{SAD} < \mathrm{sqrt}(\mathrm{C}[\mathrm{W}_{\mathrm{DCT}}]\sum(\mathrm{UB}_{|\mathrm{F-F'}|}[\mathrm{QP}][\mathrm{W}_{\mathrm{DCT}}])^2) \qquad (4-28)$$

결론적으로, 식(4-28)의 우변이 ZB detection을 위한 SAD의 upper-bound인 것이다.

$$UB_{SAD}[QP][W_{DCT}] = sqrt(C[W_{DCT}] \quad \sum(UB_{|F-F'|}[QP][W_{DCT}])^2) \qquad (4-29)$$

SAD가 없고 SATD만이 유효한 경우에 있어서는, SATD에 대한 upper-bound는 식(4-29)로부터 유도된다. HEVC에서는 motion estimation에 대한 중간 결과들이 주로 SATD로 표현되기 때문에, SAD보다는 SATD가 유효할 경우가 상대적으로 많다. 연구[6]에서는 잔차 신호를 Gaussian distribution으로 모델링하고 픽셀간의 correlation을 이용하여, SAD에 대한 threshold를 SATD의 threshold로 바꾸는 방법을 제안하고 있다. 연구[6], 이러한 확률 모델을 통해 SAD에 대한 threshold와 SATD의 threshold 간의 일정한 상수비를 유도하였고, 본 논문에서는 이 상수비를 활용하여 SATD에 대한 upper-bound를 정하였다. 아래 식에서 $UB_{SATD}[QP][W_{DCT}]$는 SATD에 대한 upper-bound를 의미한다.

$$UB_{SATD}[QP][W_{DCT}] = 1.72 \times UB_{SAD}[QP][W_{DCT}] \qquad (4-30)$$

4.2.4 Proposed ZB detection for HEVC

앞서 설명된 조건들을 기반으로, HEVC를 위한 ZB detection알고리즘이 설계되었고, 이는 그림 4.4에 요약되어 있다. 첫 번째 단계로는, 주어진 블록의 SATD를 식(4-30)의 $UB_{SATD}[QP][W_{DCT}]$와 비교한다. 해당 조건을 만족하지 않으면, 입력 블록은 ZB가 아닌 것으로 판단한다. 이 조건을 만족하는 블록에 대해서, DCT의 크기에 따라 그 다음 과정이 진행된다. 4x4와 8x8 블록에 대해서는 line 3의 if 조건문으로 이동하고, 16x16과 32x32에 대해서는 line 5의 else 조건문으로 이동한다. 4x4와 8x8 블록에 대해서는 식(4-10)의 threshold와 비교를 하여, 모든 Hadamard coefficient가 threshold보다 작을 때, 해당 블록을 ZB으로 검출한다. 이런 기능이 line 3에서 수행된다. 16x16과 32x32에 대해서는 line 6에서 GZB을 위한 식(4-12)의 조건문이 수행되고 line 8에 대해서는 PZB를 위한 식(4-15)의 조건문이 수행된다. 최종적으로, line 10에서 해당 블록이 GZB 또는 PZB이면 해당 블록을 ZB로 검출한다.

```
1 if ( SATD < UB_SATD[QP][W_DCT] ) {
2   if (W_DCT = 4 or W_DCT = 8){
3      if (NUM_hadamard= 0){ ZB = 1;}
4   }
5   else {
6     if(NUM_hadamard= 0 && DC_hadamard = 0) //GZB condition in (12)
7              GZB = 1;
8     else if (NUM_hadamard_x2 = 0 && NUM_4x4 = 0) //PZB condition in (15)
9              PZB = 1;
10    ZB = GZB or PZB;
11  }
12}
```

그림 4.4 HEVC를 위해 제안된 ZB detection algorithm

그림 4.4의 알고리즘은 기본적으로 Inter-prediction mode의 Luma data를 위해 설계되었다. 그러나, Merge mode의 Luma data와 모든 Chroma data의 경우에는 RDO전에 FME를 수행하지 않기 때문에, SATD가 유효하지 않다. 따라서, 이러한 경우에 대해서는 SATD를 구하기 위해서는 Hadamard transform을 추가로 수행해야 하는데, 이로 인한 연산량의 증가가 무시할 만큼 작지 않다. 그러므로, 이러한 경우에 대해서는 연산량이 적은 SAD를 SATD 대신 활용하여, SAD와 식(4-29)에서 제공하는 SAD의 upper-bound와 비교하는 것으로 제안 알고리즘의 첫번째 단계를 수정하도록 한다. 이를 통해, 제안된 ZB detection 알고리즘의 연산의 복잡도를 크게 줄일 수 있다.

4.3 Evaluation

이번 섹션에서는 새로 제안한 ZB detection 알고리즘의 효과를 측정하여 제시하고자 한다. 알고리즘의 효율성과 정확도를 측정하기 위해 FNR(false negative ratio)와 FPR(false positive ratio)를 사용하며, 이는 아래와 같은 식들로 표현 된다.

$$\text{FNR} = \frac{\text{the number of ZBs misclassified as Non-ZBs}}{\text{the number of ZBs}} \quad (4-31)$$

$$\text{FPR} = \frac{\text{the number of Non-ZBs misclassified as ZBs}}{\text{the numbe of Non-ZBs}} \quad (4-32)$$

FNR이 낮을수록 ZB detection 알고리즘을 통해 줄일 수 있는 연산량이 늘어나며, FPR이 낮을수록 잘못 ZB로 결정되는 블록의 수가 줄어들기 때문에 RD 저하를 감소시킬 수 있다. 이 두 개의 지표를 활용하여, ROC (receiver operating characteristic) curve를 구할 수 있고, 이런 ROC curve와 최적의 FNR과 FPR로 구성된 최적점(0,0)까지의 최소거리로 ZB detection 알고리즘의 성능을 평가하도록 한다. 그러나, FNR이 크면 줄어는 연산량이 작아질 뿐 RD 저하에는 영향이 없지만, FPR이 크면 RD 저하에 영향을 주기 때문에, 최소거리를 측정할 때, FPR이 0.1을 넘지 않는 구간에서 측정하였다.

첫 번째 실험은 제안된 ZB detection 알고리즘과 기존의 알고리즘의 성능을 비교하는 실험으로, 알고리즘의 성능뿐만 아니라 연산의 복잡도의 비교를 위해 실행시간에 대한 차이도 비교하고 있다. 실험 조건은 low-

delay configuration을 사용하였으며, Transform depth를 3으로 설정하였다. Class B (1920x1080) 영상 4개와 Class C (832x480) 영상 4개를 사용하였고, FNR,FPR 그리고 ROC curve와 (0,0)까지의 최소거리의 실험결과를 각 Class의 영상 4개에 대한 평균치로 아래 표.3에서 보여주고 있다. 4x4 와 8x8에 대해서는 기존 알고리즘이 좀 더 짧은 최적점까지의 최소거리를 제시하고 있지만, 그 차이는 매우 작다. 반면에, 16x16 과 32x32에 대해서는, 기존의 알고리즘에 비해, 새로 제안한 알고리즘이 훨씬 좋은 성능을 보이고 있다. 16x16에 대해서는 새로 제안한 알고리즘이 FNR을 480p 영상과 1080p 영상 각각에 대해서 50.6% 과 59.8%만큼 감소시켰다. 또한, 32x32에 대해서는 새로 제안한 알고리즘이 FNR을 480p 영상과 1080p 영상 각각에 대해서 27.3% 과 44.5%만큼 감소시켰다. 이런 차이의 가장 근본적인 이유는 기존 알고리즘은 4x4와 8x8 같은 작은 블록을 대상으로만 제안되어서, 같은 조건을 16x16 과 32x32에 적용했을 때, PZB를 제대로 검출할 수 없기 때문이다. 이러한 FNR의 차이는 결국 최적점까지의 ROC curve의 최소거리에도 영향을 주게 된다. 평균적으로, 이러한 최소거리의 차이는 16x16과 32x32 에 대해서, 각각 0.198과 0.203이다.

표 4.3 기존 알고리즘과 새로 제안된 알고리즘의 성능 비교

		Extended Algorithm			Proposed Algorithm		
Resolution	DCT	FNR	FPR	Distance	FNR	FPR	Distance
480p	4x4	0.080	0.011	0.081	0.085	0.010	0.086
	8x8	0.186	0.013	0.186	0.198	0.011	0.198
	16x16	0.453	0.045	0.457	0.224	0.037	0.228
	32x32	0.656	0.012	0.657	0.477	0.014	0.477
1080p	4x4	0.049	0.020	0.053	0.050	0.019	0.054
	8x8	0.134	0.029	0.137	0.137	0.027	0.140
	16x16	0.331	0.079	0.341	0.133	0.098	0.173
	32x32	0.533	0.025	0.533	0.296	0.055	0.306
Average	4x4	0.065	0.016	0.067	0.068	0.015	0.070
	8x8	0.160	0.021	0.162	0.168	0.019	0.169
	16x16	0.392	0.062	0.399	0.179	0.068	0.201
	32x32	0.595	0.019	0.595	0.389	0.035	0.392

다음 실험은 새로 제안된 알고리즘과 기존 알고리즘을 수행하는데 걸리는 시간을 비교하였다. 아래 그림.5는 기존 알고리즘의 수행시간 대비 새로 제안된 알고리즘의 수행시간을 보여주고 있다. 여기서 언급하는 알고리즘의 수행시간은 Hadamard transform과 thresholding을 수행하는 데 걸리는 시간을 의미한다. 이러한 알고리즘 수행시간이 측정되어 비교되고 있다. 480p 영상에 대해서는 수행시간의 비가 34.4% 와 38.9% 사이에 분포하고, 1080p 영상에 대해서는 수행시간의 비가 37.0%와 38.1% 분포한다. 이러한 결과, 새로 제안된 알고리즘의 연산의 복잡도가 기존 알고리즘의 연산의 복잡도보다 모든 영상에서 40% 미만임을 확인할 수 있다.

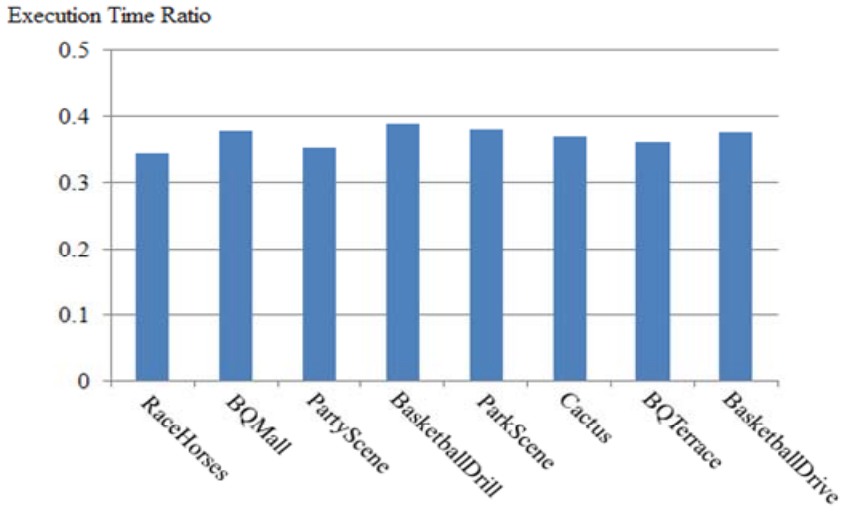


그림 4.5 기존 알고리즘과 새로 제안된 알고리즘의 수행시간비

ZB detection에 의해 RDO 이전에 ZB으로 검출된 블록에 대해서는 RDO 과정을 수행할 필요가 없기 때문에, 해당하는 연산량이 줄어들게 된다. 다음실험은 제안된 알고리즘의 RDO 수행시간의 감소와 RD 저하의 정도를 확인하기 위해 진행되었다. 이 실험을 위해, 제안된 알고리즘을 모두 Luma data 뿐만 아니라 Chroma data까지 모두에 적용하였으며, 성능 비교를 위해 기존 알고리즘에도 같은 조건으로 실험하였다. RDO 수행시간의 측정을 위해, RDO의 과정인 DCT, Q, IQ, IDCT와 CABAC의 수행시간이 각각 측정되었다. HM의 기존 RDO를 기준으로 하여 RDO 수행시간의 감소와 BD-BR의 증가가 측정되었다. RDOQ는 quantization process에 영향을 주기 때문에, 이 실험에서는 해당 옵션을 끄고 실험하였으며, 이후에 RDOQ의 옵션을 켠 상태로도 실험을 따로 진행하였다. 기존 알고리즘도 HM에 구현되어, RDO 수행시간의 감소와

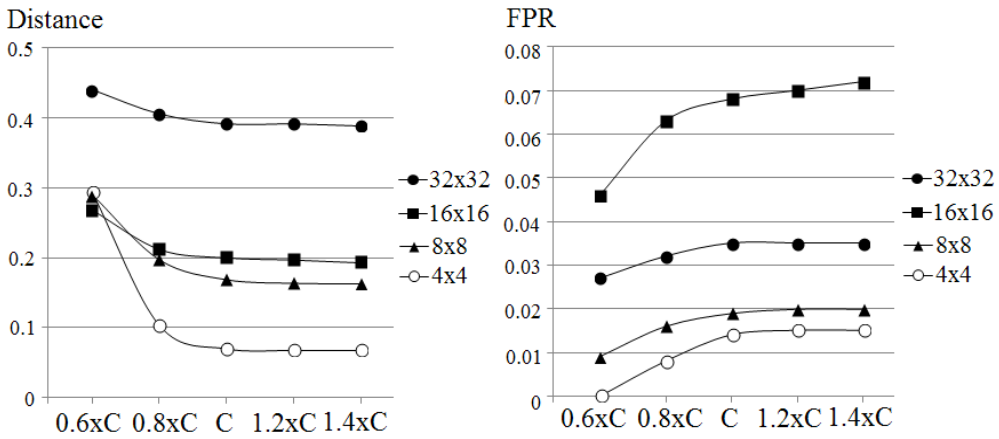
BD-BR의 증가가 따로 측정되어, 아래 표 4.4에 새로 제안된 알고리즘의 결과와 같이 제시되어 있다. 기존 알고리즘과 새로 제안된 알고리즘의 BD-BR의 증가가 각각 0.22%와 0.33%로, 두 알고리즘 모두 RD 저하가 매우 작은 것에 반해, RDO 연산량의 감소는 새로 제안된 알고리즘의 훨씬 큼을 알 수 있다. 새로 제안된 알고리즘이 평균적으로 RDO의 연산을 17% 더 감소시킬 수 있다.

표 4.4 기존 알고리즘과 새로 제안된 알고리즘의 적용시, BD-BR 증가와 Time Saving 비교

		Extended Algorithm		Proposed Algorithm	
Resolution	Sequence	BD-BR increase	Time Savging	BD-BR increase	Time Saving
480p	RaceHorses	0.10%	11%	0.24%	24%
	BQMall	0.35%	23%	0.55%	39%
	PartyScene	0.22%	15%	0.34%	31%
	BasketBallDrill	-0.19%	23%	-0.17%	37%
1080p	ParkScene	0.18%	30%	0.37%	50%
	Cactus	0.20%	25%	0.24%	43%
	BQTerrace	0.08%	33%	0.12%	56%
	BasketBallDrive	0.78%	23%	0.92%	40%
	Average	0.22%	23%	0.33%	40%

추가되는 실험은 새로 제안된 ZB detection 알고리즘에 사용되는 파라미티의 값의 변화에 따른 성능 변화를 측정하기 위해 수행되었다. 아래 그림은 식(4-22)의 C를 다양하게 바꾸면서, ROC curve와 최적점까지의 최소거리를 보여주고 있다. 그림 4.6(a)는 에서의 x축의 "0.6xC"는 원래 C의 값에 0.6을 곱한 것을 의미하며, 비슷한 규칙으로,

"0.8xC","1.2xC"" 그리고 "1.4xC"들은 각각 원래 C의 값에 0.8, 1.2 그리고 1.4를 곱한 것을 의미한다. 이 그림에서 보듯이, 최적점까지의 최소 거리의 감소가 C 근처에서 포화됨을 알 수 있다. 반면에, 그림 4.6(b)에 보듯이 C가 증가할수록 FPR이 증가함을 알 수 있다. 이러한 FPR의 증가는 ZB detection 알고리즘이 피해야할 RD 성능의 저하를 발생하기 때문에, 새로 제안된 알고리즘에서는 C를 사용하도록 한다.

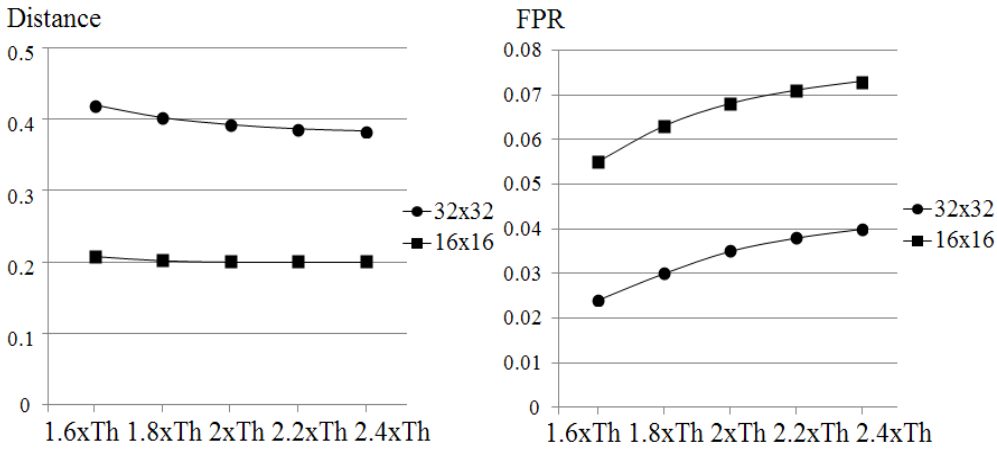


(a) Minimum distance (b) FPR

그림 4.6 식(4-22)의 C의 변화에 따른 알고리즘의 성능 변화

다음 실험은 식(4-15)의 PZB 를 위한 threshold의 변화에 따른 ZB detection 알고리즘의 성능의 변화를 측정하기 위해 수행되었다. threshold의 값은 "1.6xTh", "1.8xTh", "2xTh", "2.2xTh" 그리고 "2.4xTh"와 같이 변화를 주어 실험하였고, 이는 아래 그림 4.7(a)와 (b)의 x축을 이루고 있다. 그림 4.7(a)는 최적점까지의 거리를 보여주고

있는데, threshold의 값이 커질수록 거리의 감소폭과 거리도 약간씩 감소함을 알 수 있다. 반면에, 앞 실험과 같이 threshold의 값이 커질수록 FPR이 증가함을 알 수 있다. 이러한 상황을 반영하여, 새로 제안된 ZB detection 알고리즘에서는 2xTh를 PZB를 위한 threshold로 채택하였다.



(a) Minimum distance (b) FPR

그림 4.7 식(4-15)의 Threshold의 변화에 따른 알고리즘의 성능 변화

섹션 4.2.3 에서, ZB detection 알고리즘의 연산의 복잡도를 줄이기 위해 SAD와 SATD의 upper-bound를 제안하고 있다. 다음 실험은 upper-bound에 의해 ZB detection 알고리즘의 연산량이 얼마나 감소하는지를 측정하기 위해 수행되었다. 아래 표 4.5는 upper-bound를 on 하고 off할 때의 알고리즘 수행시간의 비를 나타내고 있다. 이 실험결과에 따르면, upper-bound를 사용함으로써 알고리즘의 수행시간을 대략 절반으로 줄일 수 있음을 알 수 있다.

표 4.5 섹션 4.2.3의 Uppper-Bound 적용시 알고리즘 수행시간비

Sequence	Exec_Time(Upper Bound On) / Exec_Time(Upper Bound Off)
RaceHorses	0.492
BQMall	0.534
PartyScene	0.496
BasketballDrill	0.556
ParkScene	0.552
Cactus	0.477
BQTerrace	0.515
BasketballDrive	0.551
Average	0.522

다음 실험은 ZB detection 알고리즘을 적용했을 때, HEVC의 전체 encoding 시간의 감소량을 측정하기 위해 수행되었다. 이 실험을 위해, ZB detection 알고리즘이 FME의 과정에 적용하여, 해당 블록이 ZB으로 판단되면, fractional motion vector를 찾는 탐색을 더 이상 수행하지 않도록 하였다. 전체 encoding 시간의 감소량과 함께 BD-BR의 증가가 함께 측정되었다. 아래 표 4.6에 해당 실험결과가 제시되어 있는데, BD-BR의 증가는 평균 1.24%로 크지 않는 데에 반해, 전체 encoding 시간은 평균적으로 12% 정도 줄어들게 된다.

표 4.6 ZB detection algorithm을 RDO와 FME에 적용시, BD-BR 증가와 전체 encoding 시간 대비 Time Saving

Resolution	Sequence	BD-BR increase	Time Saving
480p	RaceHorses	1.64%	7%
	BQMall	1.27%	13%
	PartyScene	0.78%	7%
	BasketballDrill	1.63%	11%
1080p	ParkScene	1.38%	15%
	Cactus	1.43%	14%
	BQTerrace	1.10%	17%
	BasketballDrive	0.73%	10%
	Average	1.24%	12%

다음 실험은 RDOQ 옵션을 on할 때의 실험으로, BD-BR의 증가와 줄어든 RDO 수행시간이 측정되었다. 아래 표 4.7에 실험결과가 제시되어 있는데, BD-BR의 증가는 평균 0.52%로 매우 작은 것에 반해, RDO 수행시간은 평균 47% 감소하였다. RDOQ 옵션을 off할 때와 비교하여, BD-BR은 약간 증가한 것에 반해, RDO 수행시간의 감소량이 평균 18% 증가하였다. 이는 ZB detection 알고리즘의 수행시간은 동일한데, ZB로 검출될 경우 RDOQ에 해당하는 수행시간까지 감소시킬 수 있어서, 전체 RDO 수행시간에 대한 감소량이 증가하기 때문이다.

표 4.7 RDOQ를 On 상태에서, ZB detection algorithm 적용시, BD-BR의 증가과 Time Saving

Resolution	Sequence	BD-BR increase	Time Saving
480p	RaceHorses	0.44%	29%
	BQMall	0.75%	45%
	PartyScene	0.54%	36%
	BasketballDrill	-0.05%	44%
1080p	ParkScene	0.58%	57%
	Cactus	0.44%	51%
	BQTerrace	0.37%	67%
	BasketballDrive	1.08%	48%
	Average	0.52%	47%

제 5 장 SATD based RDO Evaluation

5.1 Difference between SSE based RDO and SATD based RDO

HEVC에서는 최적의 mode를 결정하기 위해서 SSE 기반의 RDO를 수행한다. 반면에 SATD 기반의 RDO는 Inter-prediction의 FME 과정에서는 최적의 fractional motion vector를 찾는 데에 이용되며, Intra-prediction에서는 35개에 해당하는 mode의 개수를 3~7개 수준으로 줄여서 SSE 기반의 RDO에 대한 입력 mode의 개수를 줄이는 데 이용된다. 이처럼 HEVC에서는 SSE 기반의 RDO의 연산량이 많기 때문에, SATD 기반의 RDO를 이를 보조하는 역할 정도로 활용되고 있다. 앞서 설명되었던 것처럼, SSE 기반의 RDO를 모두 SATD 기반의 RDO로 대체하여 최적의 mode를 선택할 경우, RD 저하의 정도가 심각하게 증가하게 된다. 그러나, SATD 기반의 RDO를 활용하여 최적의 mode를 결정할 경우 SSE 기반의 RDO에 비해 연산량이 매우 작기 때문에 연산량이 크게 줄어 들게 된다. 이러 이유로, SATD 기반의 RDO를 활용하여 최적의 mode를 정하는 것은 여전히 매력적인 방법이다. H.264/AVC에서는 RDO를 on/off 할 때의 RD저하가 상대적으로 작기 때문에, SSE-RDO를 SATD-RDO로 부분적으로 대체하는 선행 연구가

없었다. 이런 이유로, 본 논문에서는, HEVC에서 SSE 기반의 RDO를 부분적으로 SATD 기반의 RDO로 대체함으로써 RD 저하를 최소화하면서 RDO의 연산량의 감소를 극대화 하는 방법을 제안하고자 한다.

이를 위해, 우선 SSE 기반의 RDO와 SATD 기반의 RDO의 차이를 설명하고자 한다. 편의상 구분을 위해, SSE 기반의 RDO는 "SSE-RDO"로 SATD 기반의 RDO는 "SATD-RDO"로 표기하기로 한다.

$$J_{SSE-RDO} = SSE + \lambda_{RDO} * Bits \quad (1)$$

$$J_{SATD-RDO} = SATD + \lambda_{SATD} * Bits \quad (2)$$

SSE-RDO의 SSE는 원신호와 재구성 신호의 차이의 제곱의 합으로 계산되며, 결국 이는 quantization error의 에너지를 의미하게 되고, SSE-RDO의 Bits는 mode, motion vector 정보와 DCT/Q의 결과를 CABAC에 적용하여 나온 bitstream의 크기를 측정하여 계산한다. 이에 반해, SATD-RDO의 SATD는 원신호와 예측 신호의 차이, 즉 잔차 신호에 Hadamard transform을 적용하여 구한 Hadamard coefficient들의 절대값의 합으로 계산하고, SATD-RDO의 Bits는 mode, motion vector 등의 정보에 대해서만 간단한 매핑이나 연산을 통해 구한다. 단, Intra-prediction의 경우는 실제 CABAC를 수행하여 나온 Bits를 적용한다. 또한, SATD는 신호의 합으로, SSE는 신호의 제곱의 합으로 연산되기 때문에, 각각의 Laglangian multiplier는 $\lambda_{RDO} = \lambda_{SATD}^2$

인 관계이다.

아래 그림 5.1은 depth가 1인 CU의 같은 잔차신호에 대한 SSE-RDO와 SATD-RDO의 RD cost의 관계를 보여주고 있다. SATD-RDO의 RD cost가 증가함에 따라 SSE-RDO의 RD cost도 증가하지만, SATD-RDO의 RD cost가 계속 커지게 되면 SSE-RDO의 RD cost의 증가가 둔화된다. 이러한 현상을 보이는 가장 큰 이유는 SATD-RDO의 SATD가 증가하여도, SSE-RDO의 SSE는 quantization 과정에 의해 그 크기를 제한 받기 때문에, SATD가 증가와 상관없이 그 상한이 정해지게 되어 증가세가 둔화되게 된다. 반대로, SATD-RDO의 RD cost가 작은 경우에는 잔차 신호에 대한 DCT/Q의 결과, 즉 quantized transform coefficient의 거의 대부분이 0이 되기 때문에, SATD가 증가하게 되면 SSE도 같이 증가하게 된다.

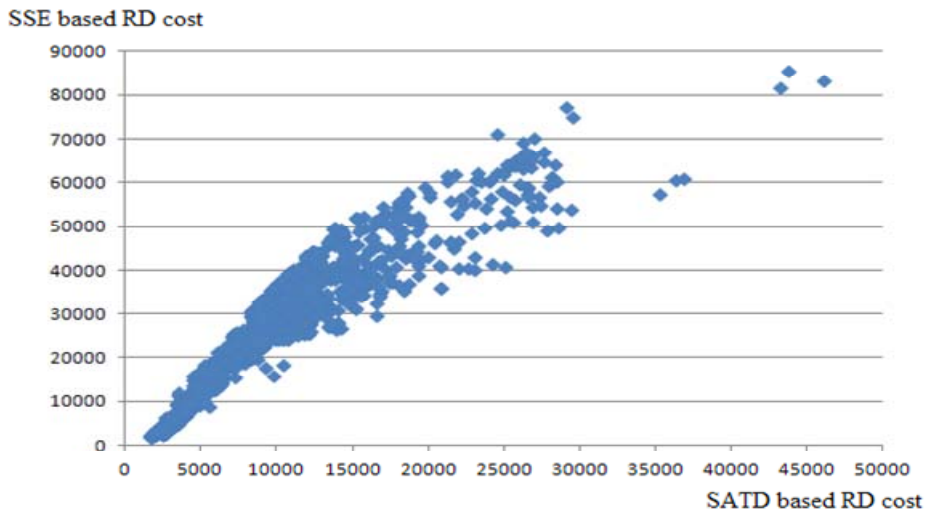


그림 5.1 SATD 기반 RD cost와 SSE 기반 RD cost의 관계

끝으로, Bits 측면에서 보자면, SATD-RDO의 RD cost가 작을 때는 SSE-RDO의 Bits중 DCT/Q에 의한 부분의 줄어들게 되어, SATD-RDO와 SSE-RDO의 Bits에 해당하는 부분이 유사하게 된다. 따라서, SATD-RDO의 Bits가 증가하게 되면, SSE-RDO의 Bits 비슷한 정도로 증가하게 된다. 반대로, SATD-RDO RD cost가 클 때는 SSE-RDO의 Bits중 DCT/Q에 대한 비중이 점차 증가하게 된다. 또한 이러한 DCT/Q에 대한 Bits의 증가는 값의 증가가 둔화되는 SSE와 결합되어, 그래프의 분포가 점점 두꺼워지는 결과를 초래하게 된다.

결론적으로, SATD-RDO의 RD cost가 상대적으로 작을 때, SSE-RDO를 SATD-RDO로 대체되면, 최적의 mode에 대한 잘못된 결정과 그 결정으로 이한 RD 저하의 정도가 상대적으로 완화됨을 예상할 수 있다.

5.2 SATD based RDO Evaluation for HEVC

SATD-RDO로 SSE-RDO를 부분적으로 대체할 때, 그룹을 짓지 않고 특정 조건에 따라 적응적으로 대체하는 방식을 따르는 경우, SATD-RDO의 RD cost와 SSE-RDO의 RD cost는 그 값의 크기가 전혀 다르기 때문에, SSE-RDO를 SATD-RDO로 대체하게 되면, 대체한 부분끼리는 SATD-RDO의 RD cost로 비교가 가능하지만, 대체한 부분과 대체하지 않은 부분끼리는 SSE-RDO의 RD cost로 비교해야 한다. 따라서, SATD-RDO의 RD cost에 따라서 적응적인 방식으로 SSE-RDO를 SATD-RDO를 대체하는 방식으로 바꾸게 되면, 효율적으로 연산을 줄이기가 어렵게 된다. 그러므로, SATD-RDO로 SSE-RDO를 RD 저하를 최소화하면서 대체할 수 있는 그룹을 찾는 방법을 제안하고자 한다.

이를 위해, SSE-RDO를 기준으로 SATD-RDO로 대체한 경우에 대해서, 때의 BD-BR의 변화를 측정하는 실험을 수행하였다. depth간의 비교, Merge mode간의 비교, Inter mode간의 비교 그리고 Intra mode간의 비교에 대해서 순서대로 SATD-RDO로 대체될 실험에는 Class C(832x480)의 영상 4개가 사용되었다. 여기서 언급되는 Merge mode간의 비교 2Nx2N Merge mode를 가리킨다. 앞서 설명된 것처럼, SSE-RDO 로 된 부분과 SATD-RDO간의 비교는 SSE-RDO의 RD cost를 비교하도록 하였다. 표 5.1은 위의 실험의 결과를 보여주고 있다. Case 1의 'SSE'는 SSE-RDO를 적용한 경우로 reference의

역할을 한다. Case 2, 3, 4 의 'Intra', 'Inter', 'Merge' 는 각각 Intra mode, Inter mode, Merge mode가 SATD-RDO로 대체된 경우를 의미한다. Case 5의 'depth CU'는 현재 CU와 하위 CU를 비교하여 CU의 depth를 결정하는 부분을 가리킨다. 끝으로 Case 6일 때, 'SATD I-Slice'는 encoding시 맨 처음 Intra frame에 대한 SSE-RDO를 SATD-RDO로 대체하는 것을 의미한다.

우선 같은 CU 레벨에서 벌어지는, Merge, Inter, Intra mode 결정과정에 대해 SATD-RDO로 대체하여 생긴 BD-BR증가는 각각 0.43%, 4.67%, 2.71% 이다. 일반적으로 Inter mode의 SATD 기반 RD cost가 Intra mode의 SATD 기반 RD cost보다 작음에도 역전 현상이 벌어진다. 그 이유는 Inter frame에서 Intra Mode의 비중이 작기 때문에 잘못 결정되어도 전체 RD 저하에 미치는 영향이 작다. 따라서, 좀 더 명확한 구분을 위해, Intra frame 만으로 encoding시 같은 CU 레벨에서의 Intra mode 결정을 SATD-RDO로 대체할 경우 BD-BR의 증가는 5.76% 로 증가하게 된다.

다음으로, CU의 depth 결정과정의 RD cost는 새롭게 결정되는 것이 아니라, CU 레벨에서 최적으로 결정된 RD cost로 결정된다. 현재 레벨 CU는 한 개의 RD cost로 대표되고, 하위 레벨 CU는 4개의 RD cost로 대표된다. 따라서, SATD 기반 RD cost 사용시 어떤 mode의 결정과정이 어느 정도 반영되는냐에 따라 RD 저하의 정도가 달라지게 된다. CU의 depth 결정과정에 SATD-RDO에 대한 BD-BR의 증가는 5.74%로 각

mode의 결정 과정들에 의한 BD-BR 증가의 합보다는 작고, Inter mode 결정과정보다 다소 큰 수준의 BD-BR의 증가를 보인다.

표 5.1 SSE-RDO를 SATD-RDO로 그룹별 대체시 발생하는 BD-BR의 증가

Case	Operation	BD-BR	BD-BR difference
1	SSE	0.00%	-
2	SSE + SATD Intra	2.71%	2.71%
3	SSE + SATD (Intra + Inter)	7.38%	4.67%
4	SSE + SATD (Intra + Inter + Merge)	7.81%	0.43%
5	SSE + SATD (Intra + Inter + Merge + depth CU)	13.55%	5.74%

이상의 실험 결과를 기반으로, SSE-RDO를 SATD-RDO로 대체하는 알고리즘을 설계하도록 하겠다. 첫 번째로는 Merge Mode의 경우 SSE-RDO를 SATD-RDO로 대체해도 RD 저하가 미미한 편이기 때문에, SATD-RDO로 대체하도록 한다. 그러나, Inter mode의 경우는 RD 저하가 크기 때문에 모두 SATD-RDO로 대체하는 것은 효율적이지 않기 때문에 다른 방법을 모색하고자 한다. HEVC에서 CU의 mode 결정순서는 "Merge mode -> Inter mode -> Intra mode" 이다. 따라서, Inter mode 결정 과정 전에, 최적의 Merge mode가 결정된다. 이 최적의 Merge mode의 SATD 기반 RD cost를 활용하여, Inter mode에 대해 SSE-RDO의 사용여부를 결정하는 방식을 제안한다. 최적의 Merge mode의 SATD 기반 RD cost를 $MERGE_{SATD\_COST}$ 로 표기하고, 현재

처리할 Inter mode의 SATD 기반 RD cost를 $INTER_{SATD\_COST}$ 로 표기하기로 하겠다. $MERGE_{SATD\_COST}$ 가 $INTER_{SATD\_COST}$ 보다 작다면, 현재 처리할 Inter mode가 최적의 mode로 선택할 확률이 낮기 때문에, SSE-RDO를 수행할 필요성이 낮아진다. 반대로 $MERGE_{SATD\_COST}$ 가 $INTER_{SATD\_COST}$ 보다 크다면, Inter mode가 최적의 mode로 선택할 확률이 높기 때문에, SSE-RDO를 수행할 필요성이 높아진다. 이를 정리하면 식(5-1)과 같이 정리된다.

$$\text{if } (MERGE_{SATD\_COST} >= INTER_{SATD\_COST}) \quad (5-1)$$
$$\text{Do SSE-RDO for Inter mode}$$

끝으로, Intra의 경우도 SATD-RDO로 전부를 대체하는 것은 RD 저하가 크기 때문에, Inter mode 에서 사용된 방법을 확장하여 사용하기로 한다. $MERGE\_INTER_{SATD\_COST}$ 는 Intra mode 결정 전에 최적 mode로 결정된 것의 SATD 기반 RD cost를 의미하고, $INTRA_{SATD\_COST}$ 는 처리할 Intra mode의 SATD 기반 RD cost를 의미한다. depth별로 RD 저하에 미치는 영향이 현격히 다르기 때문에 depth에 따라 가중치를 다르게 설정하였다. CU의 depth가 2 이하 일 때는 W[depth] 는 1로, depth가 3 일 때는 W[depth] 는 2로, depth가 4 일 때는 W[depth] 는 4로 설정된다. 결론적으로, Intra mode 에 대해서는 식(5-2)와 같이 표현된다.

if ($MERGE\_INTER_{SATD\_COST}$*W[depth] >= $INTRA_{SATD\_COST}$) (5-2)

Do SSE-RDO for Intra mode

이상의 제안된 알고리즘을 정리하면, Merge mode에 대해서는 모든 SSE-RDO를 SATD-RDO로 대체한다. 단, SATD-RDO로 결정된 최적의 Merge mode에 대해서는, Inter 또는 Intra mode 와도 RD cost 비교를 해야 되기 때문에, SSE-RDO를 수행한다. Inter mode와 Intra mode에 대해서는 식(5-1)과 식(5-2)의 조건을 만족할 때만 SSE-RDO를 수행한다. 이렇게 되면, CU 레벨에 대해서 최소 한 번 이상의 SSE-RDO를 수행하기 때문에, CU depth 를 결정하기 위해, 별도의 SSE-RDO를 수행할 필요는 없다.

5.3 Evaluation

이번 섹션에서는 섹션 5.2 에서 제안된 SATD based RDO Evaluation을 적용할 경우, 줄어드는 RDO 연산량과 RD 저하의 실험 결과를 제시하고자 한다. 측정을 위해, Class B (1920x1080) 영상 4개와 Class C (832x480) 영상 4개를 사용하였고, low-delay configuration으로 encoding 하였고, RDOQ는 on 하였다. 줄어드는 RDO 연산량을 측정하기 위해, 전체 encoding 수행시간을 측정하였고, 제안된 알고리즘을 적용하여, 기준 대비 줄어든 encoding 수행시간을 TS(Time Saving)으로 나타내고, RD 저하를 표시하기 위해 BD-BR의 증가를 표기하였다.

표 5.2 Merge Mode의 SATD-RDO 대체와 식(5-1), (5-2) 누적 적용시 BD-BR의 증가와 Time Saving

		Merge		Merge+Inter		Merge+Inter+Intra	
Resolution	Sequence	BD-BR	TS	BD-BR	TS	BD-BR	TS
480p	RaceHorses	0.52%	2.38%	0.83%	2.79%	1.15%	4.16%
	BQMall	0.66%	4.92%	1.00%	6.39%	1.15%	9.05%
	PartyScene	0.48%	4.32%	1.06%	5.37%	1.07%	8.50%
	BasketballDrill	0.82%	3.92%	1.02%	5.12%	1.55%	7.19%
1080p	ParkScene	0.85%	3.65%	1.49%	4.86%	1.55%	5.80%
	Cactus	0.93%	3.69%	1.45%	4.95%	1.62%	5.93%
	BQTerrace	0.96%	5.04%	1.48%	6.80%	1.55%	8.06%
	BasketballDrive	1.19%	1.52%	1.61%	1.98%	2.21%	2.34%
	Average	0.80%	3.68%	1.24%	4.78%	1.48%	6.38%

위의 표 5.2는 섹션 5.2에서 제안된 방법을 적용했을 때의 실험

결과를 보이고 있다. 'Merge'는 Merge mode 결정 과정을 SATD-RDO로 대체한 것이고, 'Inter'는 식(5-1)의 조건을 적용한 것을, 'Intra'는 식(5-2)의 조건을 적용한 것이다. 'Merge' 인 경우 BD-BR의 증가는 평균 0.80% 이고 줄어든 encoding 수행시간 감소는 평균 3.68% 이다. 'Merge+Intra'의 경우는 BD-BR의 증가는 평균 1.24% 이고 줄어든 encoding 수행시간 감소는 평균 4.78% 이다. 모든 mode 결정과정에 적용한 'Merge+Intra+Inter'의 경우는 BD-BR의 증가는 평균 1.48% 이고 줄어든 encoding 수행시간 감소는 평균 6.38% 이다.

제 6 장 결론

본 논문에서는 HEVC에서의 RDO 연산을 줄이면서 RD 저하를 최소화할 수 있도록 크게 세가지 방향의 연구들을 통해, 다양한 방법을 제안하였다.

첫 번째로는, RDO의 각 과정에 속하는 DCT/Q/IQ/IDCT 그리고 CABAC를 줄이는 기술들을 소개하였다. Simplified SSE는 H.264/AVC에서 제안된 기술로, HEVC에서도 쉽게 적용되며, 수학적인 관계를 이용하여 IDCT를 생략하는 방법을 제안하고 있다. 수학적인 관계를 기반으로 설계되었기 때문에 RD 저하가 거의 없다. Simplified CABAC으로는 세 가지 복잡도로 H.264/AVC에서 제안된 연구들을 HEVC에 적용하였다. High 와 Medium-complexity CABAC은 CABAC의 주요 과정을 단순화하는 기술을 제안하였고, Low-complexity CABAC는 CABAC의 주요 과정을 거치지 않고, Symbol과 Bits 사이의 통계적인 관계를 이용하여 Bits의 량을 예측하였다. Low-complexity CABAC는 HEVC에 적합하도록 수정되었고, non-zero quantized transform coefficients만을 이용하는 Low-complexity CABAC을 제안하였다. 끝으로는, DCT/Q의 과정을 거치지 않고, RDO를 수행할 수 있는 Threshold Algorithm과 Simplified SSE & CABAC without transform을 제안하였다. 이상의 기술들을 통해 BD-BR의 증가가 평균 5.13% 이고 encoding 수행시간 감소 7.1% 인 조합과 BD-BR의 증가가 평균 5.93% 이고 encoding 수행시간 감소 7.8% 인

조합을 제안하였다. 이 조합들의 BD-BR 증가 수준은 H.264/AVC의 RDO를 on/off했을 때의 BD-BR의 증가 수준과 비슷하다.

두 번째로는, HEVC에 적합한 Zero Block detection 알고리즘을 제안하였다. HEVC에서는 H.264/AVC에서 고려할 필요가 없었던, Pseudo ZB가 존재하게 되기 때문에, 이를 검출할 수 있는 Zero Block detection 알고리즘이 필요하게 된다. 새로 제안된 ZB detection 알고리즘을 통해, RDOQ off시 RDO의 수행시간을 평균 40%, RDOQ on시는 47% 감소시키며, 각각의 BD-BR의 증가는 0.33% 와 0.52%이다. 또한 제안된 알고리즘은 FME에도 쉽게 적용이 가능하여 Search Points 개수를 큰 RD 저하 없이 줄일 수 있다. RDO와 FME에 모두 적용했을 경우, 전체 encoding 시간은 평균 12% 감소하며, BD-BR의 증가는 1.24%이다.

세 번째로는, SSE 기반의 RDO를 SATD 기반의 RDO로 부분적으로 대체하는 방식을 제안하였다. 일반적으로 SATD 기반의 RDO는 SSE 기반의 RDO에 비해 연산량이 작고, Distortion과 Bits로 계산하는 값이 다르기 때문에, SSE 기반의 RDO에 비해 그 정확도가 낮다. SATD 기반의 RDO로 대체되어도 RD 성능 저하에 영향이 작은 경우가 존재하기 때문에, 이러한 점을 활용하여, Merge mode에만 적용하면, BD-BR 증가는 평균 0.80% 이고, RDO의 수행시간은 평균 3.68%, 줄어든다. Merge mode의 SATD 기반 RD cost를 활용하여 Inter와 Intra mode의 RDO를 조건부로 수행하게 되면, BD-BR 증가는 평균 1.48% 가 되어 증가하지만, RDO 수행시간은 평균 6.38%까지 감소하게

된다.

이상 세 가지 방향으로 제안된 기술들은 다음과 같은 특징이 있다. Simplified RDO는 RDO 각각의 과정을 단순화하는 기술인데, 조건에 따라 적응적으로 적용하지 않고, 해당 과정을 모든 경우에 대해서 단순화하는 기술이다. 여러 기술들을 조합할 경우, RD 저하가 증가하게 되나 항상 일정 비율의 연산량을 줄이는 데 용이하다. 따라서, 최악의 경우를 반영하여 설계를 하는 Hardware 구현에 적용할 경우, Hardware의 크기를 일정하게 줄 일수 있는 장점이 있다. ZB detection은 RDO의 연산을 줄이면서도, RD 저하가 거의 없는 매우 효율적인 기술이다. 그런데, 영상과 QP에 따라서 ZB의 비율이 달라지 때문에 ZB detection에 의한 RDO 수행시간의 감소량 또한 유동적이다. 그러나, ZB detection 알고리즘은 RDO 뿐만 아니라 FME의 연산을 줄이는 데에도 효율적인 사용이 가능한 것처럼 ZB detection을 적용을 통해, encoding 전체의 시간을 줄이는 데에 확정 적용이 용이하다. SATD 기반의 RDO 부분적 사용으로 Merge mode의 경우는 일정하게 RDO 연산을 줄이는 데에 반해, Inter와 Intra mode의 경우는 cost 비교 조건에 따라 RDO 연산이 유동적으로 줄어들게 된다. 앞의 두 기술의 일정성과 유동성을 동시에 지니면서, RD 저하가 크지 않기 때문에, Hardware와 Software 구현 모두에 효과적이다.

끝으로, 이상의 세 방향의 기술을 같이 사용함으로써, RDO의 연산량을 줄이면서 RD 저하를 최소화할 수 있다. 기술의 적용 범위를 세세하게 나누면 그 조합의 수가 매우 크게 증가함으로, 본 논문에서는 각각

연구에서 제시하는 알고리즘 중 줄어드는 연산량 대비 RD 저하가 가장 작은 기술들의 조합하여 최적 알고리즘을 설계하였다. 최적의 알고리즘은 Simplified SSE, ZB detection 그리고 SATD based RDO 로 구성된다. 이 알고리즘의 조합의 성능은 HM에 이미 채택된 fast encoder algorithm중에 하나인 ECU[28]와 비교 분석되었다. ECU는 Early CU decision을 의미하는 것으로, 현재 CU가 Skip mode로 결정되면, 하위 CU를 처리하지 않는 알고리즘이다. 표 6.1은 BD-BR의 증가와 줄어든 encoding 시간을 표기하고 있다. ECU 적용시 줄어든 encoding 시간은 31.96%이고 BD-BR의 증가는 0.00%로 매우 효과적인 결과를 얻을 수 있다. 이에 반해, 제안된 조합 알고리즘 적용시, 줄어든 encoding 시간은 14.08%이고 BD-BR의 증가는 2.14%로 ECU에 비해 성능이 떨어짐을 확인할 수 있다. 그러나, 본 논문의 알고리즘은 RDO의 연산을 줄이는 것이기 때문에, ECU와 호환되어 사용이 가능하다. 따라서, ECU를 켠 상태로, 제안된 알고리즘을 적용하면, 줄어든 encoding 시간은 39.05%로 증가하고 BD-BR의 증가는 2.22% 수준을 유지할 수 있다.

표 6.1 제안된 조합 알고리즘과 ECU 적용시 BD-BR 증가와 Time Saving

		ECU		Proposed Algorithm		ECU+ Proposed Algorithm	
Resolution	Sequence	BD-BR	TS	BD-BR	TS	BD-BR	TS
480p	RaceHorses	0.08%	15.8%	1.84%	10.0%	1.83%	21.9%
	Bqmall	0.08%	33.4%	1.96%	18.4%	1.90%	43.4%
	PartyScene	0.03%	21.4%	1.85%	17.6%	1.84%	33.5%
	BasketballDrill	-0.13%	29.5%	1.64%	15.8%	1.66%	38.1%
1080p	ParkScene	0.16%	40.8%	2.31%	13.2%	2.83%	46.0%
	Cactus	0.16%	35.8%	2.17%	13.5%	2.65%	40.9%
	BQTerrace	-0.24%	48.6%	2.24%	16.9%	2.01%	55.6%
	BasketballDrive	-0.11%	30.4%	3.13%	7.2%	3.03%	33.0%
	Average	0.00%	31.96%	2.14%	14.08%	2.22%	39.05%

표 6.2는 제안된 조합 알고리즘을 RDO off 상태와 비교한 결과를 보여주고 있다. RDO off 에 의해 줄어든 encoding 시간은 21.6%로 제안된 조합 알고리즘에 비해 50% 정도 크지만, BD-BR의 증가가 17.23%로 제안된 조합 알고리즘에 비해 8배 이상 증가하기 때문에, 제안된 조합 알고리즘이 훨씬 효율적으로 encoding 시간을 감소시키며 RD 저하를 최소화함을 알 수 있다.

표 6.2 제안된 조합 알고리즘과 RDO off 적용시 BD-BR 증가와 Time Saving

		RDO off		Proposed Algorithm	
Resolution	Sequence	BD-BR	TS	BD-BR	TS
480p	RaceHorses	13.45%	22.8%	1.84%	10.0%
	Bqmall	16.33%	24.0%	1.96%	18.4%
	PartyScene	11.82%	23.3%	1.85%	17.6%
	BasketballDrill	20.44%	23.1%	1.64%	15.8%
1080p	ParkScene	14.95%	20.7%	2.31%	13.2%
	Cactus	20.15%	20.8%	2.17%	13.5%
	BQTerrace	17.22%	22.2%	2.24%	16.9%
	BasketballDrive	23.51%	16.2%	3.13%	7.2%
	Average	17.23%	21.6%	2.14%	14.1%

참고문헌

[1] L.-M. Po and K. Guo, "Transform-Domain Fast Sum of the Squared Difference Computation for H.264/AVC Rate-Distortion Optimization," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 17, no. 6, pp. 765 773, June 2007.

[2] J. Hahm and C. M. Kyung, "Efficient CABAC Rate Estimation for H.264/AVC Mode Decision," *IEEE Trans. Circuits and Syst. for Video Technol*, vol. 20, no. 2, pp. 310 316, Feb. 2010.

[3] J. Hahm, J. Kim and C. M. Kyung, "A Fast CABAC Rate Estimator for H.264/AVC Mode Decision," in *Proc. of Int. Conf. on Acoustics, Speech and Signal Processing ,* pp. 929-932, 2009.

[4] L. Liu and X. Zhuang, "Cabac Based Bit Estimation for Fast H.264 RD Optimization Decision," in *Proc. IEEE Int. Conf. Consumer Communication and Networking Conference,* pp.1-5, 2009.

[5] Frank Bossen, "CE1: Table-based bit estimation for CABAC", JCTVC-G763, JCT-VC Meeting, Geneva, November 2011.

[6] JCT-VC, "WD5: Working Draft 5 of High-Efficiency Video Coding", JCTVC-G1103, JCT-VC Meeting, Geneva, November 2011.

[7] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification, Joint Video Team, ITU-T Recommendation H.264 and ISO/IEC 14496-10 AVC, May 2003.

[8] G. J. Sullivan and T.Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 11, pp. 74-90, Nov. 1998.

[9] Joint Video Team of ITU-T VCEG and ISO/IEC MPEG, Joint Model Reference Software, Version 18.0.

[10] ISO/IEC-JTC1/SC29/WG11, *HEVC reference software manual*, ed. Geneva, Switzerland, 2011.

[11] G. Bjonteggard, "Calculation of average PSNR differences between RD-curves, the 13th VCEG-M33 Meeting, Austin, TX, Apr. 2001.

[12] JCT-VC, "Common test conditions and software reference configurations", JCTVC-G1200, JCT-VC Meeting, Geneva, November 2011.

[13] Q. Chen and Y. He, "A fast bits estimation method for rate-distortion optimization in H.264/AVC," the Picture Coding Syst. (PCS 2004), paper no.35, Dec. 2004.

[14] S. Chen and S. Sun, "Efficient Bit-Rate Estimation Technique for CABAC," in *Proc. IEEE Int. Conf. Asia Pacific Conference (APCCAS),* pp.514-517, 2008.

[15] J. L. Nunez Yanez, V. A. Chouliaras, "Hardware Assisted Rate Distortion Optimization with Embedded CABAC Accelerator for the H.264 Advanced Video Codec," *IEEE Trans. Consumer Electron.*, vol. 52, no. 2, pp. 590-597, May 2006.

[16] ISO/ICE MPEG and ITU-T VCEG, "Working draft number 2, revision 8 (WD-2 rev 8)," JVT-B118r8, April 2002.

[17] Y.-W. Huang, B.-Y. Hsieh, S.-Y. Chien, S.-Y. Ma, and L.-G. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 4, pp. 507 522, Apr. 2006.

[18] X. Zhou, Z. Yu, and S. Yu, "Method for detecting all-zero DCT coefficients ahead of discrete cosine transformation and quantization," *Electron. Lett.*, vol. 34, no. 19, pp. 1839 1840, Sep. 1998.

[19] Z.-G. Xie, Y. Liu, J. Liu, and T. Yang, "A general method for detecting all-zero blocks prior to DCT and quantization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 2, pp. 237 241, Feb. 2007.

[20] Z. Liu, L. Li, Y. Song, S. Li, S. Goto and T. Ikenaga, "Motion Feature and Hadamard Coefficient-Based Fast Multiple Reference Frame Motion Estimation for H.264," *IEEE Trans*

Circuits Syst. Video Technol.,vol. 18, no.5, pp.620-632, May 2008.

[21] JCT-VC, "Simplification on CBF inference and coding", JCTVC-J0241, JCT-VC Meeting, StockHolm, Jul. 2012.

[22] H. Wang and S. Kwong, "Prediction of Zero Quantized DCT coefficients in H.264/AVC Using Hadamard Transformed Information," *IEEE Trans Circuits Syst. Video Technol.*,vol. 18, no.4, pp.510- 515, Apr. 2008

[23] JCT-VC, "HM8.0 Common Test Conditions and Software Reference Configurations", JCTVC-J1100, JCT-VC Meeting, Geneva, StockHolm, Jul. 2012.

[24] JCT-VC, "HEVC reference software manual", JCTVC-J0470, JCT-VC Meeting, StockHolm, Jul. 2012.

[25] H. Wang and S. Kwong, "Hybrid model to detect zero quantized DCT coefficients in H.264," *IEEE Trans. Multimedia*, vol. 9, no. 4, pp.728 735, Jun. 2007.

[26] H. Wang and L. A. Sousa, "General method for eliminating redundant computations in video coding," *Electron. Lett.*, vol. 36, no. 4, pp. 306 307, Feb. 2000.

[27] H. Wang, S. Kwong, and C. W. Kok, "Analytical model of zero quantized DCT coefficients for video encoder optimization," in *Proc*. IEEE ICME'06, July 2006, pp. 801-804.

[28] K. Choi, S.-H. Park, and E. S. Jang, Coding Tree Pruning Based CU Early Termination, document JCTVC-F092,JCT-VC, Torino, Italy, Jul. 2011

ABSTRACT

The increasing popular demand for multi-media devices such as Full-HD TV, Smart-Phone and Tablet PC motivates the new compression standard for high definition video. High Efficiency Video Coding (HEVC) is the next generation video compression standard developed by ISO/IEC MPEG and ITU-T/VCEG. In January 2013, Final Draft International Standard (FDIS) is published and then the standardization process is complete. The HEVC standard aims to improve the coding efficiency over H.264/AVC by a factor of two. To this end, several new compression tools including a flexible block structure, the intra-coding with new spatial prediction directions and sophisticated interpolation filters are adopted in HEVC. Specially, the flexible block structure and the intra-coding with new spatial prediction directions is the most important factor for the enhancement of the coding efficiency. For this reason, Rate-Distortion Optimization (RDO) to decide the best mode becomes more important whereas the flexible block structure including various sizes of transforms increases dramatically the computational complexity of RDO. Therefore, sustaining the coding efficiency with the reduction of RDO computation becomes the important issue in HEVC.

In this paper, RD degradations for HEVC and H.264/AVC with RDO on/off are compared with the experimental result and the three categories of

algorithms to reduce the RDO computation are proposed. The first category of proposed algorithms is to simplify the computation complexity for the partial process of RDO such as Transform, Quantization, Inverse Quantization, Inverse Transform and Entropy Coder. Previous research for H.264/AVC becomes the basis of these algorithms and analyzed for more aggressive algorithms. The second category of proposed algorithms is the Zero Block detection algorithm targeted for HEVC. Previous algorithms of Zero block detection for H.264/AVC have no consideration the characteristics of Zero Block in HEVC. Thus, the performance of previous algorithms is limited. To overcome this limitation, the new Zero Block detection algorithm is proposed for HEVC. The third category of proposed algorithms is SATD based RDO utilization. Basically, SSE based RDO presents less RD degradation and more computation than SATD based RDO. However, for the specific case, SATD based RDO might provide the similar coding efficiency. To this end, the difference between SATD based RDO and SSE based RDO is analyzed and based on the analysis, the efficient SATD based RDO is proposed. All of proposed algorithms are implemented in HM, HEVC reference software and, the experimental result shows the dramatic reduction of RDO computation and a negligible RD degradation.

KeyWords: **HEVC, RDO, Simplified CABAC, Simplified SSE, Zero block detection, Pseudo zero block, SATD upper bound, SATD based RDO**

Student Number: **2009-30926**

감사의 글

박사 논문 심사를 끝내고 나서, 잠시 여유를 두며 뒤를 돌이켜 보니 많은 과정들이 있었습니다. 그때마다 어떻게 하지의 걱정과 꼭 해내야지의 열정이 뒤엉키며 앞으로 한 걸음씩 나아갔던 것 같습니다. 그 한 걸음 한 걸음을 혼자가 아닌 여러 분들의 도움 덕분에 멈추지 않았기에 오늘의 기쁨을 얻을 수 있었습니다. 학위 논문 끝에 그 분들께 이런 감사의 글을 남길 수 있어서 기쁘게 생각합니다.

오랜 세월 동안 자식이 가는 길을 응원해주시고 기다려 주신 부모님께 감사 드립니다. 학부 졸업 이후 지금까지 결코 짧지 않은 시간 동안 묵묵히 지원해주신 덕분에 연구의 길을 갈 수 있었습니다. 앞으로도 좋은 모습 보일 수 있도록 노력하겠습니다. 다음으로, 어린 나이에 나와 결혼해준 아내에게 감사의 말을 남기고 싶습니다. 2000년에 처음 만나 오늘까지의 긴 시간 동안, 무채색 같은 사람에게 유채색의 아름다움을 느끼게 해줘서, 내 삶을 풍성하게 해줘서 고맙습니다. 좋은 남편의 모습을 보일 수 있게 애쓰겠습니다.

연구의 A부터 Z까지 가르쳐 주신 지도 교수님께 감사 드립니다. 처음 연구실에 들어 왔을 때의 저와 지금의 저를 비교해 보니 많이도 성장했다는 생각이 듭니다. 의욕만 앞선 저에게 날카로움과 까다로움을 갖출 수 있게 해주셔서 고맙습니다. 그러나, 아직도 부족한 점이 많다는 것을 잘 알기에, 그 가르침을 밑거름 삼아 좀 더 나은 사람으로 성장할

수 있도록 최선을 다하겠습니다.

끝으로, 연구실의 선, 후배 들에게 고맙다는 말을 전하고 싶습니다. 미국에서의 유학생활을 정리하고 한국에 돌아왔을 때, 다시 잘해낼 수 있을까의 두려움이 저를 누르고 있었습니다. 그러나, 연구실 동료들 덕분에 쉽게 벗어날 수 있었습니다. 저의 개성을 받아주신 선배들께 감사드리며, 저의 엄격함을 인내해준 후배들께 고맙다는 말을 전하고 싶습니다. 제가 받은 것만큼이나 동료들에게 보답하고자 했지만 많이 부족했던 것 같습니다. 넓은 이해를 부탁 드립니다.